CHIP SPECIAL

19,– DM

COMPUTER STEUERT MÄRKLIN-MODELL-BAHN

Anwendungen für Märklin-Digital-HO ● Hardware ● Software ● Fahrstraßen ● Automatikbetrieb ● Modellzüge simulieren Prozesse ● "train-ing" – ein System für die Ausbildung

Programme für

- **Atari ST**
- **Amiga**
- **Commodore 64/128**
- **IBM PC und Kompatible**
- **und alle Computer mit serieller Schnittstelle**

CHIP SPECIAL

82005/87001
28,– DM

BESSER PROGRAMMIEREN

C

Der schnelle C-Einstieg

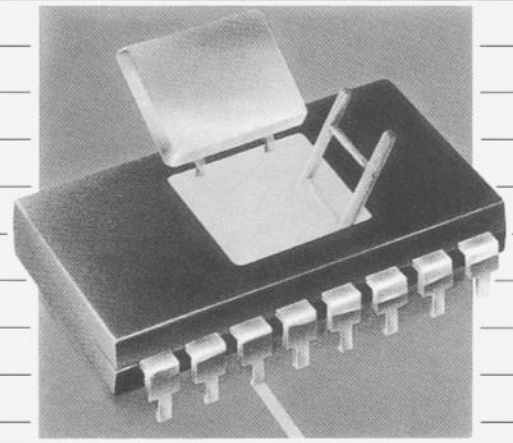
Mein erstes Programm
Umgang mit Konstanten
Alle C-Operatoren
Elementare Datentypen
Anweisungen in C
Funktionen aufrufen
Die vier Speicherklassen
Zeiger und Vektoren
Nützliche Strukturen
Der C-Präprozessor
Wichtige Routinen
Arbeiten mit Syntaxdiagrammen

Mit Anwendungsbeispielen für
MOS-DOS: Lattice-C, Manx-Aztec-C, Microsoft-C
CP/M: Manx-Aztec-C, BDS-C
Atari ST: Megamax-C, Lattice-C
Amiga: Manx-Aztec-C
Macintosh: Megamax-C

Helmut Ostermann

Maschinen- und Assemblersprache des M68000

Eine Einführung mit vielen Beispielen



CHIP WISSEN

CHIP SPECIAL

82005/87010
28,– DM

68000

User-Guide zum Prozessor 68000

Atari ST, Amiga, Sinclair QL

- Die 68000-Familie im Überblick: 68008, 68010, 68020, 68030
- Floatingpoint-Unit
- Integer-Arithmetik
- Schnelle Verschiebe-, Sortier- und Suchroutinen
- Peripherie-Bausteine

Alle Programme auf Diskette erhältlich

CHIP SPECIAL

82005/87012
28,– DM

Ausgabe 7

PASCAL

Turbo-Pascal

Programmier-Praxis

- Textanalyse und -verarbeitung
- Dynamische Strings
- HGC-Toolbox
- Bausatz Editor
- Präsentationsgrafik
- Automatisches Stichwortverzeichnis
- Neue Benutzeroberfläche – selbst gestaltet

Alles über Turbo-Pascal 4.0

Sämtliche Programme für alle MS-DOS- und CP/M-Computer auf Datenträger erhältlich

Der richtige Anschluß

Sie fasziniert nicht nur, sondern trägt inzwischen auch wesentlich zum Verständnis für die Informations-, Steuerungs- und Regelungstechnik bei: Die computergesteuerte Modelleisenbahn.

Im vorliegenden ERSTEN MÄRKLIN-SPECIAL wird am Beispiel Märklin-Digital-HO in Verbindung mit Programmen für vier Rechnermodelle aufgezeigt, was sich mit der Kombination Computer und Modelleisenbahn alles steuern und regeln läßt.

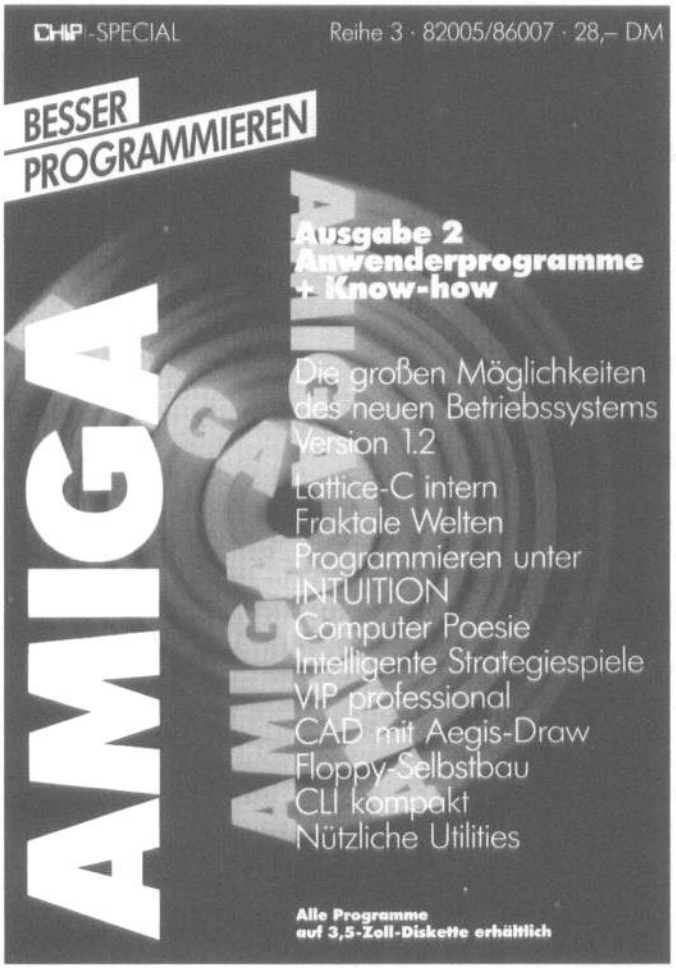
Die erste mit Digitaltechnik gesteuerte Modellbahnanlage stellte die Gebr. Märklin & Cie. GmbH bereits auf der Nürnberger Spielwarenmesse 1980 vor. Der Autor dieser Ausgabe, Digitalfreak der ersten Stunde, packte - mit Hilfe des Programmierteams von CHIP-SPECIAL - in diese Ausgabe all seine Erfahrungen, die er mit dem Gespann Eisenbahn & Computer im Laufe der Jahre gesammelt hat. Gerhard Bader zeigt nicht nur auf, wie das Digital-HO-System im Zusammenspiel mit dem Computer funktioniert, sondern gibt konkrete Entscheidungshilfen. Lohnt sich für mich das Umsteigen auf Digitaltechnik? Wie gehe ich vor, wenn ich meine bestehende Anlage umbauen will? Welche Möglichkeiten sind mit welchem Aufwand verbunden?

Soviel vorweg: Es lohnt sich, Schritt für Schritt - wohlüberlegt vorzugehen. Und so richtig Spaß macht es erst, wenn man weiß "weshalb was wie" funktioniert! Dieses Heft trägt seinen Teil dazu bei. Zudem erfahren Sie, was sich mit dem Lehr- und Lernsystem 'train-ing' alles machen läßt. In Zusammenarbeit mit den Kultusministerien von Niedersachsen, Nordrhein-Westfalen, Rheinland-Pfalz und Baden-Württemberg wird es im Schulunterricht eingesetzt. Durch die Verbindung von Computer und digitalisierter Modelleisenbahn kann der Informatikunterricht auf eine anschauliche, attraktive Weise praxisbezogen gestaltet werden.

Bevor Sie ans Werk gehen, noch eine - zumindest für uns - bemerkenswerte Information, die aus einer Studie des Heinrich-Bauer-Verlages stammt: In unserem Lande besitzt bereits jedes 5. Mädchen im Alter zwischen sechs und vierzehn Jahren eine Modelleisenbahn.

Viel Freude beim Umgang mit Computer und Eisenbahn wünscht Ihnen

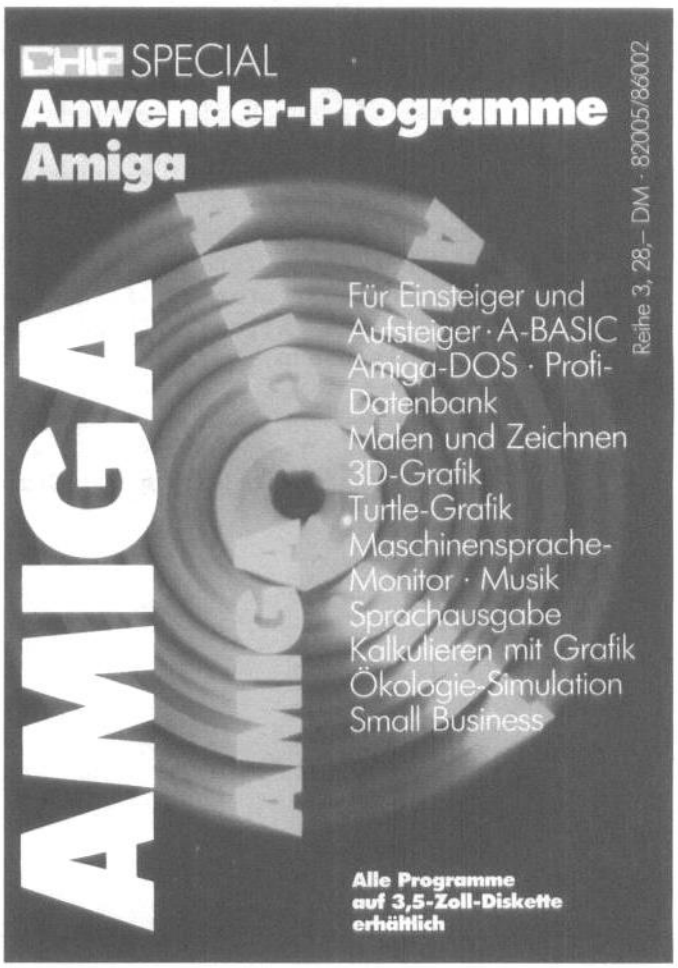
Ihre Redaktion CHIP-SPECIAL



In dieser Ausgabe

Autor: Reiner Kunz

In dieser Ausgabe



Inhaltsverzeichnis

CHIP SPECIALS

Das Fischer-Technik-Interface schafft die Verbindung zwischen verschiedenen Computersystemen und den Schaltern, Sensoren, Motoren und Magneten der Fischer-Technik-Baukästen. Was sich damit alles verwirklichen läßt, wird anhand von nachvollziehbaren Beispielen und Programmen aufgezeigt, die vier Rechner unterstützen (ST, C64/128, CPC und IBM PC). Mit drei unterschiedlichen Software-Lösungen wird beispielsweise ein Roboter aufgebaut, der als Rückmeldesystem Schalter, Potentiometer und Lichtschranken benutzt.

Best.-Nr.: 0952, DM 19,–

Die computergesteuerte Modelleisenbahn fasziniert nicht nur Väter, sondern trägt inzwischen im Schulunterricht zum Verständnis für die Informations-, Steuerungs- und Regelungstechnik bei. Was sich mit der Kombination Computer und Modelleisenbahn alles steuern läßt, wird hier am Beispiel Märklin-Digital-HO in Verbindung mit Programmen für C64/128, Atari ST, Amiga, IBM PC und Kompatible und allen Computern mit serieller Schnittstelle aufgezeigt. Das hier vorgestellte Lehrmittelsystem „train-ing" wird übrigens bereits erfolgreich in Haupt- und Realschulen, Gymnasien und Berufsschulen im Unterricht eingesetzt. Und zwar in Zusammenarbeit mit den Kultusministerien von Niedersachsen, Nordrhein-Westfalen, Rheinland-Pfalz und Baden-Württemberg.

Best.-Nr. 0953, DM 19,–

BESTELLCOUPON:

Bitte ausfüllen, unterschreiben und einsenden an CHIP-Leser-Service 735, Vogel-Verlag, Postfach 6740, D-8700 Würzburg 1

Ja, bitte liefern Sie mir die angekreuzten Produkte zu den genannten Preisen plus Versandkosten.

	Stück	Best.-Nr.	Einzel-Preis DM
Fischer-Technik und Computer		0952	19,–
Computer steuert Märklin-Modelleisenbahn		0953	19,–

Datum, Unterschrift

Vorname, Name 1451

Straße, Nr.

PLZ, Ort

Die Lieferung der Specials erfolgt gegen Rechnung plus Versandkostenanteil.

SOFORT BESTELLEN

Start

Seit der Modellbahn-Pionierzeit versuchten Konstrukteure oder auch Hobby-Tüftler, den kleinen Bahnen möglichst viele Eigenschaften des großen Vorbilds mitzugeben. Teilweise gelang ihnen das so gut, daß die Abbildungen geschickt fotografierter Modellbahnen oft erst bei genauer Betrachtung von Großbetriebsaufnahmen zu unterscheiden sind. Doch nicht nur optisch, sondern auch in Form vieler Betriebsmöglichkeiten kam man der Realität sehr nahe. Sanftes Beschleunigen oder Abbremsen, Betriebsaufenthalte, Pendelverkehr und zahlreiche andere Raffinessen lassen sich jetzt schon mit elektronischen Fahrpulten realisieren. Dennoch blieben bisher einige entscheidende Wünsche unerfüllt, wie unabhängiger Betrieb möglichst vieler Triebfahrzeuge ohne Trennstellen, konstante Zugbeleuchtungen, Schaltung von Sonderfunktionen an Loks und Wagen sowie der Aufbau auch ausgedehnter Weichenstraßen ohne das bisher übliche störanfällige Kabelbäume.

Bereits auf der Nürnberger Spielwarenmesse 1980 stellte Märklin die erste digitalgesteuerte Modellbahnanlage vor. Die Erfahrungen aus dieser Prototyp-Anlage bildeten die Grundlage für die Entwicklung des serienreifen Systems.

Zu dem schon verwirklichten unabhängigen Betrieb mehrerer Triebfahrzeuge auf demselben Gleis mit gemeinsamer Übertragung von Energie und Steuersignalen sowie der Steuerung aller Weichen über eine einzige 2polige Anschlußleitung kamen bei der endgültigen Produktkonzeption zwei weitere entscheidende Forderungen hinzu:

Die Empfängerelektronik muß auch in die kleinsten Märklin-HO-Loks passen.
Das neue System muß mit bestehenden Märklin-Anlagen kompatibel sein, d.h., der Märklin-Modellbahner soll seine Märklin-Artikel in gewohnter Weise weiterverwenden können. Ferner dürfen die neuen Digital-Triebfahrzeuge auf konventionellen Anlagenteilen keinen Fahrbeschränkungen unterliegen, sondern sollen sich im Falle eines etappenweisen Um- oder Aufbaus einer Modellanlage freizügig in der "alten Wechselstromwelt" bewegen können.
Dieser letzte Punkt erwies sich im Laufe der Vorarbeiten als die "härteste Nuß", die erst durch die Entwicklung eines Custom IC gelöst werden konnte. Dieser "Märklin-Chip" enthält etwa 10 000 Transistorfunktionen.

Märklin Digital HO

Die Beschäftigung mit der Modelleisenbahn fasziniert. Eine komplexe Anlage mit Drehscheibe, Gleisharfe, Ablaufberg, mehrstreckigem Betrieb zwischen verschiedenen Bahnhöfen und einem möglichst Schattenbahnhof, um Zuggarnituren austauschen zu können, steht sicher auf der Wunschliste vieler. Wer jedoch elektrische Weichen, verschiedene Stromkreise und das ganze noch mit Rückmeldung zu Gleisbildstellwerken betreiben will, kommt sehr schnell zu einem "Drahtverhau". Zu jedem Magnetartikel führt mindestens ein in der Regel jedoch zwei Steueranschlüsse, mit denen eine Weiche von Geradeaus-Fahrt aus abzweigen, ein Signal von rot auf grün gestellt wird. Die Notwendigkeit, möglichst viele Funktionsgruppen elektromagnetisch zu steuern und möglichst noch über Rückmeldeeinrichtungen Bescheid zu wissen, wo Lokomotiven und Züge sich tatsächlich befinden, hat im Laufe der Zeit dazu geführt, daß zunächst Relaisschaltungen, dann Digital-Elektronik-Schaltungen zu entwickeln, um beispielsweise Schattenbahnhöfe vollautomatisch zu steuern oder mit Signalen und Kontaktgleisen Blockstrecken automatisch zu betreiben. Neben dem simplen Schienennetz, das bei Märklin aus dem Dreileitersystem besteht, stand ein ganzes Netzwerk aus Steuer- und Rückmeldeleitungen, die dafür sorgten, daß der Modellbahnbetrieb wirklichkeitsnah und problemlos ferngesteuert werden konnte. Einen wesentlichen Schritt nach vorn bringt das Digital-HO-Konzept, das Märklin zum 100jährigen Bestehen vorstellte und das inzwischen über alle wesentlichen Komponenten verfügt, um mit und ohne Computer realistischen Fahrbetrieb zu realisieren. Einiger "Drahtverhau" ist geblieben, um Magnetartikel an Decoder und Rückmeldemodule anzuschließen. In Vorbereitung befinden sich jedoch bereits Weichen und Signale, die so wie die Lokomotive über einen integrierten Decoder verfügen, so daß eigentlich nur noch die Leitungen von den Rückmeldebausteinen verlegt werden müssen. Der gesamte andere Modellbahnbetrieb kann, wenn in Kürze digitalisierte Weichen und Signale zur Verfügung stehen, über zwei Adern gesteuert werden.

Digital HO funktioniert seriell

Da nur zwei Leitungen zur Verfügung stehen, mit denen sowohl die Versorgungsspannung als auch die Information an die Betriebsstrecke gelegt werden, werden Informationen seriell übertragen. Im Gegensatz zu manchen anderen Digitalschaltungen wird bei Digital HO jedoch nicht zwischen den zwei Zuständen Spannung oder Nicht-Spannung hin- und hergeschaltet. Um den problemlosen Betrieb der Lokomotiven unabhängig von der Informationsübertragung sicherzustellen, wird zwischen einer positiven und negativen Spannung hin- und hergeschaltet.

Da alle Informationen für alle Verbaucher oder Triebfahrzeuge über das zweiadrige Netz der Gleisanlage geschickt werden, müssen die Informationen unterteilt, also in einen Adress- und Datenteil aufgesplittet werden. Es ist festgelegt, aus wievielen Einzelinformationen der Adressteil und der Daten- oder Informationsteil besteht. Der Adressteil besteht bei Digital HO aus 4 bzw. 5 Bits, der Befehlsteil aus 5 bzw. 4 Bits, je nachem ob Lokomotiven oder Magnetartikel gesteuert werden. Der Namensteil für Lokomotiven besteht deshalb aus 4 Bits, da 5 Bits benötigt werden, um Fahrgeschwindigkeit, Fahrtrichtung und Sonderfunktionen zu steuern. Rein rechnerisch ergibt sich nach dem Dualsystem die maximale Anzahl von 16 Lokomotivadressen. Da Märklin Digital HO nicht binär sondern trinär arbeitet, ergeben sich 3 hoch 4 Möglichkeiten, das sind 81. Eine Kombination wurde als sogenannte Nullinformation ausgeblendet. Bei den Weichen und Signalen besteht der Namensteil aus 5 Bits - das ergibt 256 Möglichkeiten. Dies rechnet sich nicht einfach nach einem Zahlsystem. Man muß wissen, daß eines der fünf Bits, den Code identifiziert und damit nur 4 Bits für die Adresse übrigbleiben. Ein Decoder wiederum kann 4 Doppelspulenantriebe steuern.

Die Lokomotiven werden über ein Fahrpult, den Control 80, Weichen und Signale, über das Stellpult, das Keyboard gesteuert. Beide Geräte einschließlich des Memory Moduls können durch das Interface ersetzt werden, das die serielle Schnittstelle zu einem beliebigen Computersystem schafft. Die beiden Steuergeräte, Control 80 und Keyboard bzw. Switchboard um Gleisbildstellwerke anzusteuern, sind über einen Mehrfachstecker über die Zentraleinheit miteinander verbunden. Die Geräte untereinander - Control 80, Keyboard, Switchboard und Memory Modul - haben ein gemeinsames Bus-System und erhalten somit Informationen, wenn ein Fahrgerät gesteuert oder über ein Stellpult Signale abgegeben werden. Die Zentralelekronik ihrerseits kann wiederum Stellpulten und Fahrpulten antworten. Die Signale werden quittiert. Über das interne Bussystem weiß jedes der Peripheriegeräte, ob der von ihm ausgesandte Befehl bei der Zentraleinheit angekommen ist und ausgeführt wurde. So ist auch sichergestellt, wenn mehrere Control 80 Fahrpulte eingesetzt werden, daß eine Lokomotivnummer nicht von mehreren Fahrgeräten aus aufgerufen werden kann. Die Zentraleinheit teilt in einem solchen Fall dem aufrufenden Fahrgerät mit, daß diese Adresse eines Triebfahrzeugs bereits belegt ist.

Um den Datentransport über das Bus-System, also die Schienen, zu optimieren, werden von der Zentralelektronik nur Informationen übertragen, wenn sich Informationen ändern. Sicherheitshalber werden Informationen zweimal hintereinander übertragen, zwischen den einzelnen Informationspaketen, einem Wort, ist eine kurze Sendepause eingeschoben, damit alle Empfänger wissen, daß nun eine neue Nachricht folgt. Der Empfänger-Baustein, gleichgültig, ob in der Lokomotive, im Decoder zur Magnetartikelsteuerung oder im Rückmeldemodul speichert die zuletzt an ihn ausgesandte Information.

Grundsätzlich hört jeder Empfänger den Anfang jedes Informationswortes mit, um festzustellen, ob der Namenadressteil ihn betrifft. Der Adressteil wird mit den über den Codierschalter eingestellten Adressen verglichen. Bei Übereinstimmung wertet der Empfänger den Datenteil aus und veranlaßt beispielsweise die Steuerung des Motors oder das Schalten eines Magnetartikels. Die Empfänger Decoder für Signale und Weichen sind aus diskreten Bauteilen aufgebaut, der Lokomotiv Dekoder mit einem Märklin-speziefischen IC, um Platz zu sparen. Die Decoder 84 und 83 bestehen aus einem Decodierbaustein, einem Codierschalter und aus Leistungstransistoren, die Ausgänge der Doppelspulen gegen Masse (= Schienenmasse=Trafomasse) schalten.

Beim Digital-HO-Konzept wurde darauf geachtet, daß von der bisherigen Konzeption nicht abgewichen wird. Das Schalten eines Signals ist weiterhin auch über ein Schaltgleis oder Stellpult möglich ist.

Wesentlich komplexer sind die Empfänger in den Lokomotiven aufgebaut. Diese Empfänger müssen folgende Eigenschaften erfüllen:

1. stufenlose Steuerung des Motors
2. Umschaltung der Drehrichtung des Motors (also die Fahrtrichtung der Lokomotive)
3. Schaltung einer Sonderfunktion, z.B. Licht oder Telex
4. Speicherung der jeweils letzten Information auch im Signalbereich
5. konventionelle Überspannungserkennung
6. Ersatz des konventionellen Umschaltrelais und der Vorschaltelektronik
7. Erkennung des Betriebssystems konventionell oder digital
8. automatische Umschaltung auf das entsprechende Betriebssystem im Millisekundenbereich.

Die gesamte Logik hierfür sitzt in einem Märklin-spezifischen Chip. Wesentlich ist die Erkennung des Betriebszustands durch einen Zähler, der in einem festdefinierten Zeitraum die an der Schiene liegenden Impulse zählt. Nur wenn die Lokomotive mit Digital HO betrieben wird, kann dieser Zähler die zur Erkennung erforderlichen 12 Impulse innerhalb von wenigen Millisekunden zählen und damit auf den digitalen Betriebszustand umschalten. Dies ist nicht ganz problemlos, da unterschiedliche Störungen, Bürsten des Motors, Prellen des Schleifers zu keinen Fehlfunktonen führen dürfen und Impulse eindeutig erkannt werden müssen. Deshalb sind aufwendige digitale Filter und Auswertstufen erforderlich. Der Lokomotiv-Decoder mit konventionellen ICs aufgebaut, benötigt den Platz von zwei eng bestückten Leiterplatten. Daß es auch einfacher geht zeigt unser Schaltungsvorschlag, der jedoch nur für Digitalsteuerung ausgelegt ist.

Die Stromversorgung für Steuergeräte, Magnetartikel und Lokomotiven erfolgt über einen Transformator, der eine Wechselspannung von 16 Volt liefert. Die Zentralelektronik selbst kann Ströme bis zu 2,5 Ampere abgeben. Bei einer Spannug von ca. 18 Volt entspricht dies einer Ausgabeleistung von rund 45 Watt. Das komplette Basisset - Zentraleinheit, ein Control 80 und ein Keyboard - hat einen Leistungsbedarf von rund 8 Watt, so daß mehr als 35 Watt zur Steuerung von Lokomotiven verbleiben. Die Zentralelektronik hat einen automatischen Abschalter, der anspricht, wenn viele Verbraucher angeschlossen sind und die Abgabeleistung von 45 Watt überschritten wird. Um noch höhere Leistungen auf die Schienen zu bringen, kann ein Booster nachgerüstet werden. Im Grundsatz können beliebig viele Leistungsverstärker eingesetzt werden, um auf großen Anlagen ausreichend Versorgungsspannung für Signale und Weichen und in unterschiedlichen Stromkreisen für die dort verkehrenden Lokomotiven bereitzustellen. Durch elektirsche Trennung der Mittelleiter können auch meherer Zentraleinheiten eingesetzt werden, um Teile einer großen Anlage autonom zu steuern. Die Lokomotiven müssen dann jedoch von einem Digitalnetz ans andere übergeben werden, wenn sie die elektrischen Trennungen überfahren. Dies kann nur ein Interface mit der entsprechenden Software leisten.

Die wesentliche Komponente des Märklin-Digital-Systems ist das Bus-System, das alle Geräte miteinander verbindet und die Kommunikaton zwischen den verschiedenen Geräten sicherstellt. Eine wichtige Komponente im Märklin-Digital-System ist das Interface, das die Aufgabe aller Steuergeräte übernimmt. Durch die entsprechenden Rückmeldemodule können über die Zentraleinheit bis zu 256 Rückmeldekontakte ausgewertet werden. In Kombination mit der Möglichkeit, 80 Lokomotiven, 256 Magnetartikel zu steuern und 256 Rückmeldekontakte auszuwerten, kann mit einem Computer über die serielle Schnittstelle der gesamte Fahrverkehr auch auf einer komplexen Anlage gesteuert werden.

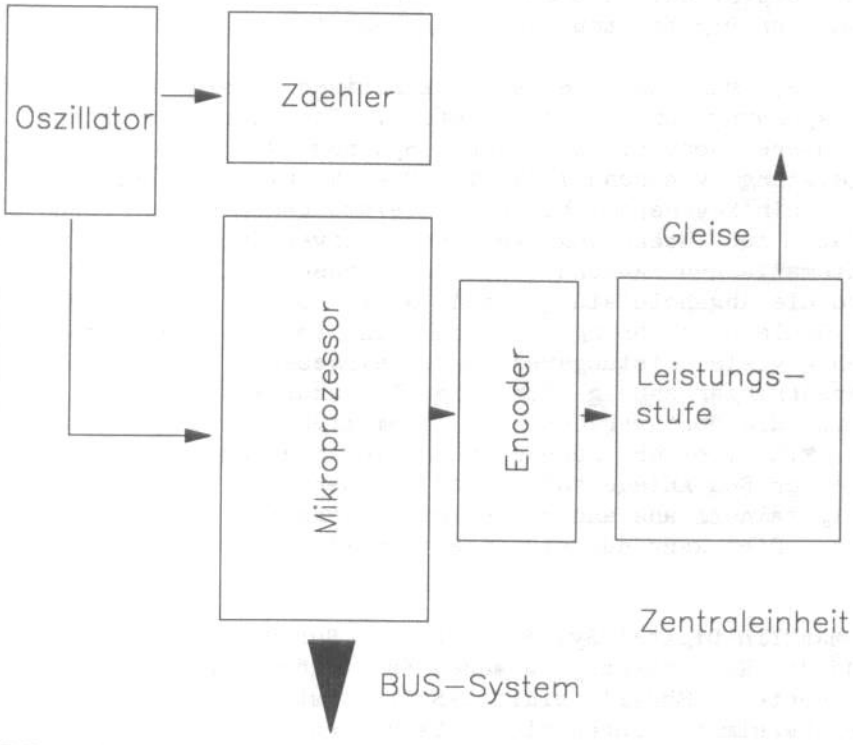
Die Komponenten des Märklin-Digital-HO-Systems

Transformator "transformer 6002"

Er erscheint in dem modernen Gewand aller Bausteine der neuen Märklin-System-Linie und weist außer zwei Paar Anschlußklemmen an der Rückseite keinerlei Bedienungselemente auf. Seine Ausgangsspannung beträgt 16 Volt, die Leistung 50 VA. Eventuell können Modellbahner auch die bisherigen HO-Beleuchtungstrafos (z.B. 6611) verwenden. Dies beschränkt allerdings die Leistung der Zentralelektronik- oder Boosterendstufen etwas. Der Transformator dient als "Kraftstation" und versorgt jeweils eine Zentraleinheit oder einen "booster" über eine zweipolige Leitung mit Strom (Anschlußklemmen "L" und "0").

Zentraleinheit "central unit" 6020

Sie bildet das eigentliche Herzstück des gesamten Systems und besitzt ebenfalls keine Bedienelemente. Lediglich eine rote Betriebsanzeige deutet dezent auf die vielfältigen Funktionen hin, die sich im Innern abspielen. Diese Leuchtdiode informiert den Benutzer außerdem über das Ansprechen der elektronischen Kurzschlußsicherung oder die am Fahrpult aktivierte Nothaltfunktion. Der eingebaute Mikroprozessor übernimmt eingehende Magnetartikelstellkommandos und schiebt sie codiert zwischen die Informationen für Triebfahrzeuge, sobald sie in die Codekette passen. Da die Übertragungsfolge rasend schnell getaktet wird, vergeht zwischen Kommandoeingabe und Signal- oder Weichenstellung weniger als ein Zehntelsekunde. Die ferner im Gehäuse enthaltene Leistungsstufe speist die codierten Signale verknüpft mit dem notwendigen Betriebsstrom über die rückseitigen Drucktastenklemmen "B" und "0" in die zweipolige Anschlußleitung zum Gleissystem.
Ebenfalls an der Geräterückseite befinden sich unter einem Kühlkörper zwei weitere Drucktastenklemmen für die 16-Volt-Versorgungsleitung vom Trafo sowie eine Steckerleiste zum Anschluß weiterer Leistungsstufen = "booster" Nr. 6015. An der linken und rechten Seitenfläche besitzt die "central unit" jeweils eine versenkt angeordnete 16polige Steckerleiste, an der sich unverwechselbar die weiteren Systemkomponenten anfügen lassen. Fahrpulte können rechts, Magnetartikelstellpulte links angereiht werden. Daher entfällt auch an der Kommandozentrale des Modell-"Fahrdienstleiters" der sonst übliche Kabelwald.



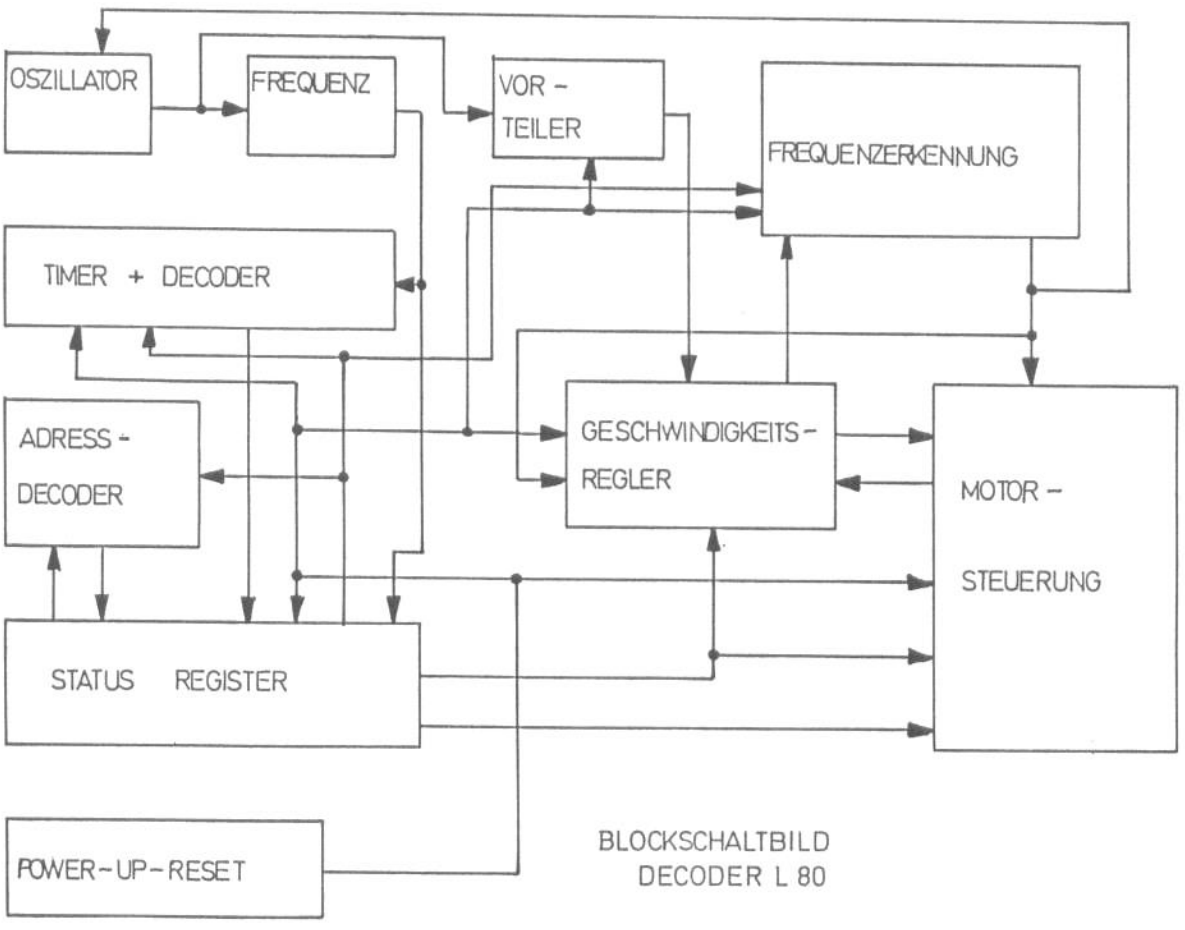
Fahrpult "Control 80" 6035

Das Fahrpult besitzt als Hauptkontrollorgan einen in gewohnter Weise bedienbaren Stellknopf. Dreht man diesen über den linken Skalenanschlag hinaus, so wird die Fahrtrichtung gewechselt. Dies entspricht völlig der Funktion von normalen Fahrgeräten, was bei stufenweiser Umrüstung von Modellanlagen die einheitliche Bedienung aller Triebfahrzeugbewegungen gewährleistet. Die entscheidenden Möglichkeiten des Fahrpults eröffnen dem Anwender aber erst zwei zusätzliche Tastenfelder. Mit zehn wie bei modernen Telefonen angeordneten Tasten von "0" bis "9" lassen sich bis zu 80 Digitaltriebfahrzeuge einzeln anwählen. Deren Empfangsbausteine kann jeder Modellbahnfreund auf beliebige "Adressen", gewissemaßen also eigene Hausnummern zwischen "01" und ..."80" einstellen. In der Regel wird man sich wohl an den Baureihennummern der Loks orientieren und beispielsweise eine Ellok der BR 103 auf "10", eine Dampflok BR 050 auf "05" oder die Diesellok BR 236 auf "23" adressieren. Die Handhabung am Fahrpult ist denkbar einfach: Zum Aufruf der auf "01" adressierten Schnellzugdampflok 012 sind lediglich die Tasten 0 und 1 zu drücken.
Danach scheint in einer Leuchtanzeige über dem Bedienfeld die Zahl 01 und die Lok

reagiert auf die Einstellung des "Fahrknebels" - nach rechts schnellere Fahrt, nach links wieder langsamer; auf "0" hält sie an. Über den Anschlag nach links gedreht, wechselt die 012er die Fahrtrichtung und fährt beim nächsten "Fahrauftrag" in der entgegengesetzten Richtung los. Während unser Schnellzug evtl. dem Bahnhof Bonn zueilt, können wir weitere Fahrzeuge in Bewegung setzen. Als nächstes rufen wir die mit "78" adressierte Tenderlok BR 78 auf. Nach dem Eintasten der Zahlen 7 und 8 steht sie sofort zur Verfügung und zieht ihre Donnerbüchsen über die romantische Nebenbahnstrecke. Dieses Spiel könnte man theoretisch fortsetzen, bis sich 80 Triebfahrzeuge auf der Anlage tummeln. Allerdings würde wohl auch der routinierteste Modellbahner bald die Übersicht verlieren, wenn für die Zugbeeinflussung kein ausgeklügeltes Signalsystem zur Verfügung steht. Dabei müssen wir bedenken, daß sich nur die jeweils aufgerufene Lok direkt beeinflussen läßt. Alle anderen Fahrzeuge bewegen sich im Sinne ihres letzten Fahrbefehls weiter. Damit die Digital-Bahner bei kritischen Fahrsituationen ein eventuell drohendes Zugunglück vermeiden können, befinden sich auf dem Fahrpult "Control 80" ferner die Nothalttaste "stop" (der gesamte Betrieb steht still) sowie die Freigabetaste "go" (alle Züge setzen sich wieder wie vorher in Bewegung, sofern die Unterbrechung nicht länger als zwei Minuten dauerte). Neben diesen Tasten stehen zwei weitere zum Ein- bzw. Ausschalten einer zusätzlichen Loksonderfunktion zur Verfügung ("function"). Mit diesen läßt sich beispielsweise die Lokbeleuchtung oder ein Rauchgenerator ein- und ausschalten (auch im Stand). Bei Telex-Loks dient die Sonderfunktion zur Fernsteuerung der Kupplung.
Tritt auf der Anlage ein Kurzschluß auf, so schaltet die Elektronik in der Zentraleinheit ab; der Betrieb wird wie bei "Nothalt" unterbrochen, bis die Störung beseitigt ist und die Elektronik durch Druck auf die Freigabetaste "go" wieder aktiviert wird. Ein Speicher im Lokempfänger ist sogar in der Lage, die letzten Fahrbefehle bei Unterbrechungen bis zu zwei Minuten zu speichern. An der linken Seite des Fahrpults befindet sich noch die 16polige Steckverbindung zur "central unit", rechts das Gegenstück zum Anfügen weiterer Fahrpulte.
Mit dem Anreihen weiterer Universalfahrpulte wächst entsprechend die Zahl der gleichzeitig steuerbaren Triebfahrzeuge. Gibt man die Adresse einer schon aufgerufenen Lok in ein anderes Fahrpult ein, so blinkt die angezeigte Adresse. Wird danach am ersten Fahrpult ein neuer Aufruf eingetastet, übernimmt das zweite Fahrpult die Kontrolle über die zuvor angewählte Lok (LED-Ziffern leuchten dann permanent).

Lokempfänger "Decoder c 80"

Dieser Baustein mißt nur 36 x 20 x 10 mm und paßt daher anstatt des konventionellen Fahrtrichtungsschalters in die Märklin-Loks. Auf der kleinen Platine befinden sich der Märklin-Chip, ein achtstelliger Codierschalter, sowie eine Reihe weiterer Bauteile in dichtester Bauweise. Dieses kleine Elektronik-Wunderwerk liest aus der Informationsflut, die die Zentralelektronik unablässig über die Schienen schickt, die an den jeweiligen Empfänger gerichteten Befehle heraus, führt sie aus und steuert gewissermaßen als unsichtbarer Lokführer Fahrgeschwindigkeit, -richtung und die Sonderfunktion direkt im Triebfahrzeug, da an den Schienen stets die volle Betriebsspannung anliegt. Dies ermöglicht den Anschluß von Beleuchtungsartikeln und motorbetriebenen Modellen (Mühlen, Pumpen usw.) an beliebigen Stellen der Gleisanlage. Damit der Lokempfänger "Decoder c 80" seine Erinnerung an den letzten Fahrbefehl auch bei längeren Stromunterbrechungen (über zwei Minuten) auf Trennstellen vor Signalen nicht verliert, ist parallel zur Fahrstromunterbrechung ein Widerstand (1,5 kOhm - 1/4W) einzusetzen. Der über diesen fließende geringe Strom dient dem Lokempfänger als "Gedächtnisstütze". Vorausschauende Digital-Aspiranten können diese Bauteile ohne Beeinträchtigung des konventionellen Betriebs schon jetzt in die Trennstellen ihrer Anlagen einbauen und ersparen sich später Nachrüstarbeiten. Der Selbsteinbau des später auch einzeln erhältlichen Lokmoduls in vorhandene Fahrzeuge stellt geübte Bastler sicher vor keine Probleme.



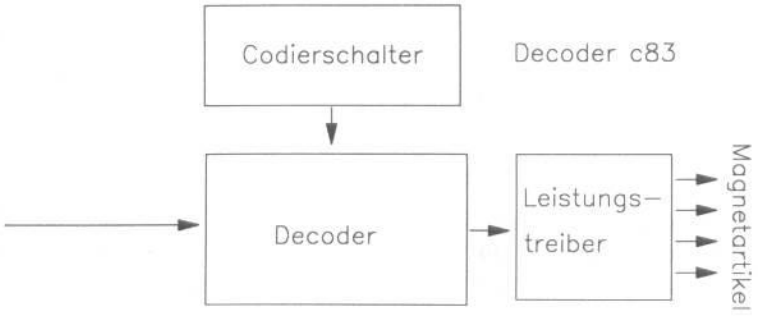
BLOCKSCHALTBILD
DECODER L 80

Magnetartikelstellpult "Keyboard" 6040

Dieses Pult wird auf der linken Seite der "central unit" eingesteckt und weist ein Bedienfeld mit 32 Tasten und 16 Leuchtdioden auf. Mit der Tastatur lassen sich bis zu 16 Zweispulen - oder (evtl. auch gemischt) maximal 32 Einspulenantriebe (z.B. Entkuppelgleise) ansteuern. Die Leuchtdioden zeigen stets "Abzweig" bei Weichen oder "Halt" bei Signalen an und erinnern den Hobby-Bahner an die zuletzt gewählte Stellung. Die Steuerbefehle werden ebenfalls über die Zentralelektronik in die Schienen eingespeist. Wem das 16er-Tastenfeld nicht reicht, der kann an der linken 16poligen Steckerreihe für weitere 16 Signale, Weichen, Bahnschranken, Entkuppelgleise usw. das nächste "Keyboard" anreihen. Bis zu 16 dieser Geräte zur Bedienung von insgesamt 256 Magnetartikeln lassen sich auf diese Weise an die Zentraleinheit anschließen. Um die Stellbefehle aus den bis zu 16 verschiedenen Pulten zu unterscheiden, befindet sich an der Rückseite dieser Systembausteine ein vierstelliger Codierschalter, der die Zuordnung von 01 bis 16 zuläßt. Damit der Besitzer von mehreren "Keyboards" seine entsprechend codierten Stellpulte nicht verwechselt, liegen diesen numerierte Klebeschildchen bei, die in ein vertieftes Feld der Pultabdeckung passen. Wird die Stromzufuhr unterbrochen bzw. die gesamte Anlage abgeschaltet, speichert das "Keyboard" die letzten Stellanzeigen.

Magnetartikel-Empfangsbaustein "decoder k 83" 6083

Dieses Modul empfängt die Steuerbefehle vom Stellpult und bezieht sowohl den Betriebsstrom als auch die Stellinformation ebenfalls über die Gleisanlage. Es ist in der Lage, bis zu 4 zweispulige (alternativ 8 einspulige) Magnetartikel anzusteuern. Damit sich der Decoder seine an ihn gerichteten Stellinformationen für die angeschlossenen vier Magnetartikel aus dem umfangreichen Datenfluß "fischen" kann, muß er ebenfalls codiert werden. Dazu läßt sich im geöffneten Gehäuse ein achtteiliger Codierschalter auf einen für jeweils vier aufeinanderfolgende Tastenpaare (1,2,3,4/5,6,7,8 usw.) gültigen Gruppencode einstellen. Am Gehäuse befinden sich neben den Anschlüssen für vier zweispulige Magnetartikel ferner zwei Buchsenpaare zur Speisung des Moduls aus der Gleisanlage und zum direkten Anschluß weiterer "decoder k 83".



Leistungsverstärker "booster" 6015

Äußerlich ähnelt dieser Baustein der Zentralelektronik. Er besitzt ebenfalls keinerlei Bedienelemente, und an der Rückseite fallen gleichermaßen die Kühlrippen ins Auge. Dagegen weisen die Seitenflächen keine Steckerleisten auf. Dieser Zusatz-Leistungsverstärker versorgt bei großen Anlagen, für die die ersten 50 VA der Kombination "Transformator/Leistungsstufe der Zentraleinheit" nicht ausreichen, einen weiteren 50-VA-Digitalabschnitt mit Energie. Dieser "booster" wird aus einem separaten Transformator gespeist, wie aus dem Blockschaltbild hervorgeht.
Unter dem rückseitigen Kühlkörper befinden sich links und rechts versenkt fünfpolige Steckerleisten für die Flachkabelverbindung zum Anschluß des Leistungsbausteins an die Zentralelektronik sowie die Weiterführung der Steuerleitung an weitere "booster". Zwei Paare Drucktastenklemmen zwischen diesen Steckanschlüssen dienen auch hier zur Versorgung des Bausteins mit der 16-Volt-Trafospannung einer- und zum Anschluß an das Gleisnetz andererseits.
Die LED-Betriebsanzeige signalisiert dem Benutzer eventuelle Kurzschlüsse auf der Anlage oder die geschaltete Nothaltfunktion.

Märklin-Digital-HO-Lokomotiven

Unter den Serienanfangsnummern 36.. liefert Märklin bereits werksseitig mit den Empfängermodulen "decoder c 80" ausgerüstete Triebfahrzeuge. Diese sind entsprechend ihrer Baureihennummern adressiert. Den Codierschalter kann aber jeder auch problemlos auf eine andere "Nummer" umstellen. An den am Empfängermodul vorhandenen Sonderfunktionsausgang ist bei normalen Triebfahrzeugen die fahrtrichtungsabhängige Spitzenbeleuchtung angeschlossen. Sie läßt sich auch im Stand ein- und ausschalten. Bei Telex-Lokomotiven bewirkt die geschaltete Sonderfunktion den Entkuppelvorgang. Ebenso könnte aber auch der Raucheinsatz von Dampfloks als Sonderfunktion geschaltet werden.

Neben diesen Grundbausteinen läßt sich das Märklin-Digital-HO-System noch weiter ausbauen:

Schnittstelle zum Computer: das "interface"

Dieses Element ermöglicht den Anschluß eines Homecomputers oder Personal-Computers an das Märklin-Digital-HO-System. Es besitzt eine serielle Schnittstelle (RS 232 mit 5-Volt und 12-Volt-Pegel).

Rückmeldemodul "decoder s 88"

Eine Automatik- und /oder Computerbetrieb erfordert die Rückmeldung der jeweiligen Zugbewegungen an die Steuerzentrale (bzw. den Computer). Dafür dienen die Rückmeldemodule. Sie übertragen jeweils bis zu 16 Meldungen von Schaltgleisen, Kontaktabschnitten, Reed-Kontakten usw. an die Steuerzentrale und lösen dadurch die nächsten Ablaufschritte eines "Programms" aus.

Decoder k 84

Während der Decoder K 83 Schaltimpulse liefert, schaltet dieser Decoder Dauerstrom, um Beleuchtungen, Motoren der Magnete dauernd mit Strom zu versorgen.

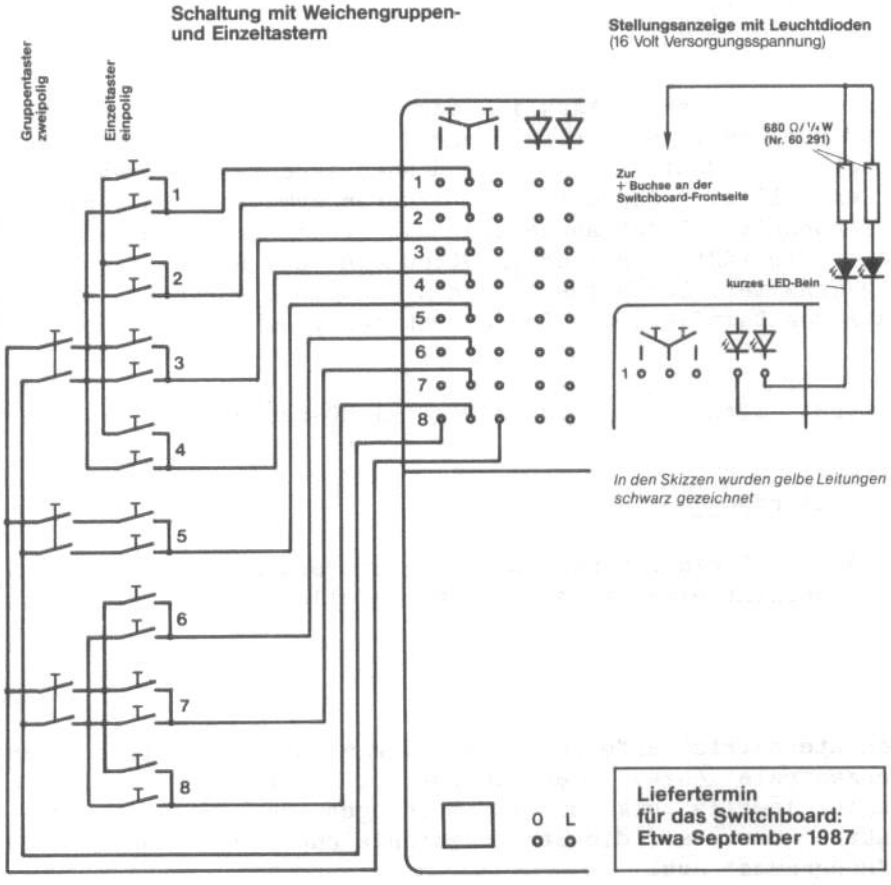
Decoder 81

Mit diesem Decoder lassen sich Lokomotiven, die einen Gleichstrommotro aben auf Dgital-HO umrüsten. Seine Funktionen entsprechen ansonsten dem Decoder c 80.

Switchboard

Das Switchboard entspricht in seinen Funktionen dem Keyboard. Es erlaubt den Anschluß von Gleisbildstellwerken. Für jede Funktionsgruppe sind die Leitungen für Schalte und Lampen herausgeführt. Dazu kommt ein CMOS-Speicher, der auch bei abgeschalteter Anlage die die zuletzt eingegebenen Schalterstellungen speichert.

Mit "Switchboard" können einzelne Schaltvorgänge (Weichen, Signale, Entkupplungsgleise etc.) über Tasten im Gleisbildstellpult ausgelöst werden. Das Switchboard entspricht also in seiner Funktion dem Keyboard. Wie beim Keyboard wird das Switchboard über einen vierstelligen Codierschalter auf eine Adresse von 1 bis 6 eingestellt.
Seiner Aufgabe entsprechend weist das Switchboard im Gegensatz zum Keyboard keine eigenen Bedienungs- und Anzeigeelemente auf. Stattdessen besitzt es Steckbuchsen, an die aber Verbindungskabel, Taster und Anzeigeleuchten angeschlossen werden können. Hierzu eignen sich die normalen Märklin-Stecker.
Für jeden Magnetartikel (Weiche/Signal) weist das Switchboard fünf Buchsen auf. Eine Dreiergruppe zum Anschluß der Momenttaster und zwei Buchsen für die Stellungsmeldung über Glühlämpchen oder Leuchtdioden.
Zum Einbau in Gleisbildstellpulte wurde das Switchboard möglichst flach ausgeführt. Es läßt sich deshalb nicht in gewohnter Weise an die anderen Digitalgeräte anstecken. Zum Anschluß ist das Adapterkabel 6038 oder 6039 vorgesehen. Dabei ist unbedingt darauf zu achten, das Switchboard nur an der linken Seite der "central unit" anzuschließen.



Anschluß der Momenttaster

Vom konventionellen System her ist man gewohnt, daß ein gemeinsames Massepotential (Masse über Momenttaster) zu den verschiedenen Verbrauchern führt.
Dazu eignen sich entweder Stellpulte 7072 oder Momenttaster in einem Gleisstellpult. Beim Switchboard-Anschluß muß man etwas umdenken. Hier wird auf einen Anschluß (in der

Mitte der Dreiergruppe) über die Taster entweder ein positives oder negatives Signal (Potential) geschaltet. Daraufhin gibt das Switchboard eine Anforderung an die Central Unit, die betreffende Weiche zu schalten. Die Central-Unit führt daraufhin den Befehl aus. Anschließend sendet sie eine Bestätigungsmeldung (Quittungssignal) zurück. Erst danach wird die Stellungsanzeige am Switchboard umgeschaltet.
Was bedeutet das für die Verdrahtung des Gleisbildstellpultes?
Für jede Weiche ist nur ein eigenes Kabel erforderlich, das jeweils an die mittlere der drei "Schaltbuchsen" führt. Die beiden Taster "rot" und "grün" werden an verschiedene Potentiale (Zuleitungen) angeschlossen (siehe Abb. "Anschluß der Momenttaster"). Es ist nicht erforderlich, dieses Potential für jede Weiche einzeln am Switchboard abzugreifen. Die Buchsen einer Reihe sind jeweils untereinander verbunden. Es reicht daher ein Anschluß je Richtung aus. Erst für die zweite Anschlußreihe (Weiche 9-16) ist eine erneute Zuleitung erforderlich.

Weichengruppentasten

Die Weichenschaltung über verschiedene Potentiale erfordert zweipolige Weichengruppentasten. Nur dadurch kann sie die Stellmöglichkeit für beide Richtungen freigeben (siehe Abb. "Weichengruppentaste"). Zusammengehörende Weichen sollten Adressen innerhalb einer 8er-Gruppe enthalten, damit sich die Anschlüsse an einer Gruppentaste vereinen lassen (z.B. Adressen 9-16 oder 33-40).

Anzeige der Weichen- und Signalstellung

Die Stellung von Weichen und Signalen wird beim Switchboard für beide Schaltzustände ("gerade" oder "rund") angezeigt. Es bietet damit mehr Anzeigemöglichkeiten als das Keyboard.

Bei der Auswahl der Anzeigeform ist man in keiner Weise eingeschränkt. Die am Switchboard angeschlossenen Anzeigeelemente erhalten ihre Energie aus einer externen Stromquelle. Hier können auch schon vorhandene Transformatoren 6631, 6671 oder 6611 eingesetzt werden. Die Spannung der externen Stromquelle darf bis zu 16 Volt betragen. Es eignet sich sowohl für Wechsel- als auch für Gleichstrom. Angeschlossen wird der Transformator an die frontseitigen Buchsen "0" und "L".
Verwendet man zur Stromversorgung einen Regeltrafo, so läßt sich die Helligkeit der Anzeige den jeweiligen Lichtverhältnissen anpassen. Es können selbstverständlich auch der gelbe und braune Anschluß des Digital-Trafos 6602 verwendet werden.
Gemeinsamer Anschluß für alle Anzeigenlämpchen oder -dioden ist die Buchse "+". Geschaltet werden sie über die zugehörigen Buchsen mit dem Leuchtdiodensymbol.
Da für die Anzeigen Gleichstrom zur Verfügung steht, erfordert der Einsatz von LEDs keine Schutzdioden. Der jeweilige Vorwiderstand von 680 Ohm ist jedoch unbedingt erforderlich (siehe Abb. "Anzeige mit LEDs").

Kombination von Switchboard, Keyboard und Memory

Durch den Anschluß an den zentralen IIC Bus (über das Adapterkabel) kann das Switchboard "mithören", welche Befehle von anderen Geräten ausgesandt werden. o ändert das Switchboard die Stellungsanzeige einer Weiche oder eines Signals auch dann, wenn sie vom Memory oder einem Keyboard geschaltet wurde. Die Umstellung der Anzeige erfolgt dabei immer erst dann, wenn die Central-Unit mit einem Quittungssignal bestätigt, daß die Schaltanforderung des Keyboards oder Memorys ausgeführt wurde. Integriert man das Memory so in das Gleisbildstellpult, so dient das Switchboard als Steuergerät für Einzelweichen und als Schaltgerät für die Ausleuchtung der über das Memory gestellten Weichenstraßen.
Das Switchboard eignet sich ebenso wie das Keyboard zur Memory-Programmierung.
Werden größere Anlagen von verschiedenen Personen bedient, lassen sich mehrere Switchboards auch auf die gleiche Adresse codieren, um z.B. gemeinsam befahrene Gleisabschnitte von allen Steuerpulten aus zu steuern und zu überwachen. Hierbei zeigen ebenfalls alle Switchboards den aktuellen Zustand auf der Anlage.

Memory

Das Memory speichert maximal 24 Fahrstraßen. Die Programmierung ist jederzeit wieder zu ändern. Jede einzelne Fahrstraße kann bis zu 20 Schaltbefehle enthalten. Sollte eine

Fahrstraße mehr als 20 Schaltbefehle erfordern, so läßt sich eine weitere Fahrstraße anhängen. Auf diese Weise sind auch lange Fahrstraßen mit einem einzigen Auslöseimpuls steuerbar. Die Schaltzeit läßt sich für jede Schaltfunktion individuell bestimmen und kann bis drei Sekunden betragen.

Sicherheit und Zuverlässigkeit sind bei der Bundesbahn die Maxima des Eisenbahnverkehrs. Durch ausgeklügelte Abhängigkeiten von Fahrweg- und Signalsteuerungen wird dieses Ziel erreicht. Schon für mittlere Bahnhöfe füllt die dazu notwendige Elektronik eigene Gebäude.

Auch bei der Modelleisenbahn bestimmen logische Abhängigkeiten zwischen Weichen- und Signalstellungen die Fahrplangestaltung, wenn nicht nur auf zwei voneinander unabhängigen Ovalen "Kreisverkehr" stattfindet.

Beispiele hierfür sind:
- die Stellung des Einfahrsignals in Abhängigkeit von der Lage der Einfahrweiche oder -fahrstraße
- die Zusammenschaltung von Ausfahrsignal und Schutzweiche oder
- Weichenkombinationen als Fahrstraßen mit Flankenschutz und entsprechenden Verriegelungen.

Die Realisierung dieser Sicherheitsschaltungen und Bedienungserleichterungen erfordern eine Fahrstraßenschaltung, d.h., die Ausführung mehrerer Schaltvorgänge auf möglichst nur einen Knopfdruck hin. Als weiterer Schritt zur vorbildgerechten Steuerung müssen auch fahrende Züge diese Funktion auslösen können. Die freizügige Einsatzmöglichkeit einer solchen Steuerung bedingt die Lösung dieser Vorgaben über eine programmierbare Steuerung, in der die Abhängigkeiten nicht durch Relais und Kabel festgelegt sind, sondern sich - jederzeit änderbar - als Programm eingeben lassen.

Diese Aufgaben übernimmt bei Märklin Didital HO entweder ein Computer, der über die serielle Schnittstelle an das Interface angeschlossen ist, oder mit eingeschränkten Progarmmiermöglichkeiten das Memory. Das Memory bietet für viele Fahrplangestaltungen ausreichende Möglichkeiten, die wir hier näher beschreiben wollen.

Anschluß an die Digital-Anlage

Das Memory wird links an die Central Unit oder an ein Keyboard angesteckt. Die Reihenfolge von Memory und Keyboard ist dabei beliebig. Es können insgesamt vier Memory angesteckt werden. An der Rückseite des Memory befindet sich ein Codierschalter, wie er auch beim Keyboard vorhanden ist. Mit diesem Schalter lassen sich beim Memory neben der Codierung (Numerierung des Memory 1 bis 4 durch die Schalter 1 und 2) die verschiedenen Betriebsarten einstellen. Darauf werden wir später noch eingehen (Schalter 3 und 4). Die 6polige Steckleiste an der Geräterückseite dient zum Anschluß von Rückmeldemodulen.

Mit den Tasten A1..A8, B1..B8 oder C1..C8 wird die jeweilige Fahrstraße durchschalten. Das geschieht als Folgeschaltung, d.h., die einzelnen Schaltvorgänge werden nacheinander ausgeführt. So belastet immer nur ein einziger Impuls die Stromversorgung der Anlage.

In eine Fahrstraße lassen sich alle Schaltvorgänge einbeziehen, die im Digital-System möglich sind (sich also sonst über ein Keyboard steuern lassen, wie z.B. Weichen, Signale, Enkupplungsgleise, Fernschalter, Schranken, Drehscheibe oder Kran).
Für teil-oder vollautomatische Betriebsabläufe können die im Memory gespeicherten Fahrstraßen auch von fahrenden Zügen ausgelöst bzw. angefordert werden. Die Impulse von Schalt- oder Kontaktgleisen oder anderen Schaltern, die einen Masseimpuls geben, werden vom Rückmeldemodul S88 aufgenommen und an das Memory weitergegeben. Diese Auslösung der Fahrstraßen von der Anlage her wird als "extern" bezeichnet. Beim Memory kann diese externe Auslösung der Schaltfunktion durch Betätigen der Taste "extern" eingeschaltet und durch Drücken der Taste "off" abgeschaltet werden.

Das Memory kann auch durch Schalter in einem Gleisbildstellwerk ausgelöst werden. Die Fahrstraßentaster werden über ein Rückmeldemodul abgefragt. Bei entsprechender Einstellung des Codierschalters am Memory werden diese Fahrstraßen zusätzlich verriegelt und sind dadurch gegen Flankenfahrten gesichert.

Die Fahrstraßen bleiben nach dem Ausschalten der Digital-Anlage im Memory weiter gespeichert.

Programmierung einer Fahrstraße

Mit dem Zusammenstecken aller Digital-Geräte bei ausgeschalteter Stromversorgung ist das Memory betriebsbereit. Nach dem Einschalten der Anlage blinken alle Leuchtdioden am Memory kurz auf. Danach leuchten nur noch die Anzeigen "extern" und "end". Drückt man jetzt die Taste "off", wird in den einfachen Betrieb umgeschaltet, d.h., das Memory schaltet nur auf Tastendruck und nicht auf Impulse, die an ein Rückmeldemodul gelegt werden.
Jede der 24 Tasten (A1 bis C8) schaltet eine ihr zugeordnete Fahrstraße (Weichen- und Signalkombination). Die Eingabe dieser Schaltbefehle erfolgt über:
- die Keyboards (digitale Stellpulte)
- das Gleisbildstellpult (das über das Switchboard mit der Digital-Steuerung verbunden ist) oder
- den Computer und das Interface.

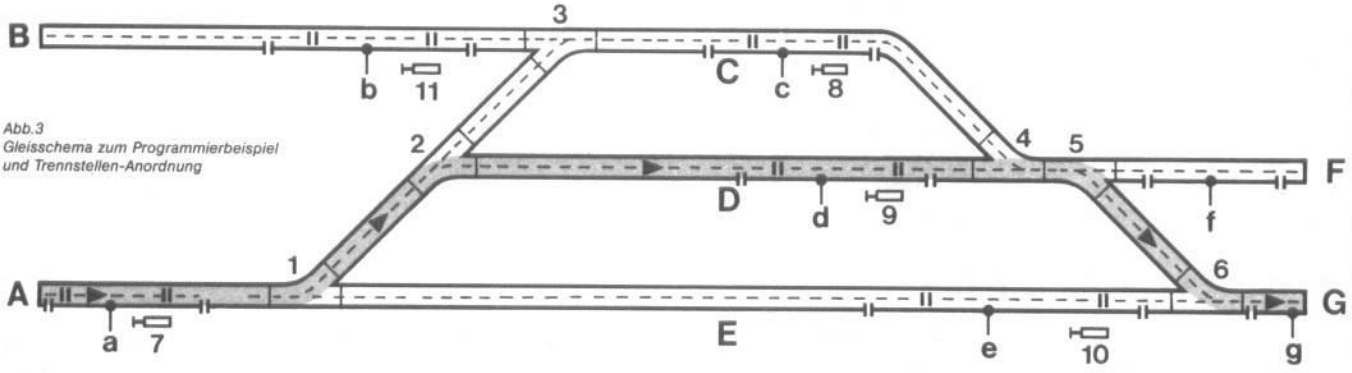


Abb.3
Gleisschema zum Programmierbeispiel und Trennstellen-Anordnung

Ein Beispiel zeigt die Programmierung

Im handgesteuerten Betrieb soll ein aus "A" kommender Zug über Weiche 1 und 2 in Gleis D einfahren. Nachdem die Reisenden ein- bzw. ausgestiegen sind, soll der Zug nach G weiterfahren. Er muß dafür Weiche 4, 5 und 6 passieren und braucht am Ausfahrsignal "Fahrt frei mit Geschwindigkeitsbegrenzung" (HP 2). Da der Zug von Hand gesteuert wird, ist es nicht erforderlich, Signal 9 zuerst in Halt-Stellung zu lassen. Die Programmierung erfolgt in folgenden Schritten als Fahrstraße "A1":
1. Am Memory Taste "input" drücken; es blinken jetzt die Leuchtdioden aller schon programmierten Fahrstraßen.
2. Am Memory Taste "A1" für die gewünschte Fahrstraße drücken; es leuchten jetzt dauernd die Dioden bei "input" und "A1".
3. Am Keyboard die Tasten: Weiche 1 rot, Weiche 2 rot, Weiche 4 grün, Weiche 5 rot, Weiche 6 rot, Signal 7 grün, Signal 9 grün drücken.
4. Am Memory Taste "end" drücken und dadurch die Eingabe beenden.

Es leuchten jetzt die Leuchtdioden bei "off" und "end".

Die Programmierung ist damit abgeschlossen.

Grundsätzlich sollte man bei Fahrstraßen die Signale zuletzt stellen. Erstens entspricht dies dem Vorbildbetrieb, bei dem das Signal erst gestellt werden kann, wenn die Weichenstraße gestellt und verriegelt ist. Zweitens sprechen auf der Modellbahn ebenfalls Sicherheitsgründe für diese Reihenfolge. Mit großer Wahrscheinlichkeit wird es zu Entgleisungen kommen, wenn zuerst das Signal auf "Fahrt frei" gestellt wird und dann erst vor dem schon fahrenden Zug die Weichen in die richtige Lage gebracht werden.

Nachdem wir die Fahrstraße "A1" einprogrammiert haben, läßt sie sich durch einmaliges Betätigen der Taste "A1" abrufen. Die Schaltvorgänge folgen rasch aufeinander. Schaltet eine Weiche oder ein Signal jetzt nicht sicher bis in die vorgesehene Endstellung, so

läßt sich die Schaltzeit für diesen Magnetartikel individuell verlängern. In unserem Beispiel soll dies für Weiche 4 der Fahrstraße "A1" der Fall sein.

Programmiert wird in folgenden Schritten:
1. Am Memory Taste "input" drücken.
   Es blinken wieder die Leuchtdioden aller schon programmierten Fahrstraßen.
2. Am Memory Taste "A1" drücken. Die Leuchtdioden bei "A1" und "input" leuchten.
3. Am Keyboard die Taste Weiche 4 grün mehrmals drücken. Die zusätzliche Schaltzeit wird automatisch an die richtige Stelle in der Fahrstraße übertragen.
4. Am Memory die Taste "end" drücken und damit die Eingabe bzw. Änderung beenden.

Jede Betätigung der Taste "Weiche 4 grün" verlängert die Schaltzeit um etwa 0,3 Sekunden. Es kann die Taste bis zu 8mal gedrückt werden. Dann ergibt sich eine maximale Schaltzeit von ungefähr 3 Sekunden (z.B. für Entkupplungsgleise). Diese verlängerte Schaltzeit läßt sich auch gleich beim ersten Einprogrammieren berücksichtigen.

Löschen einer Fahrstraße

Einprogrammierte Fahrstraßen kann man auch auf einfache Weise wieder löschen. Anstelle der Eingabe von Weichenstellungen im Programmierschritt 3 wird die "clear" Taste sooft betätigt, bis die Anzeige von "input" auf "end" umspringt. Alle Schaltbefehle dieser Fahrstraße sind jetzt gelöscht. Das Löschen der Fahrstraßen erfolgt von hinten nach vorne, d.h. der zuletzt einprogrammierte Befehl wird zuerst gelöscht. Es ist also auch möglich, nur die letzten Befehle einer Fahrstraße zu löschen, um Änderungen vorzunehmen. Fahrstraßen lassen sich auch durch die Tastenfolge "input", jeweilige Fahrstraße "A1...C8" und "clear" löschen.

Der Betrieb mit Verriegelung

Will man eingestellte Fahrstraßen gegen das Stellen einer kreuzenden Weichenstraße sichern, so läßt sich das Memory auf die Betriebsart "mit Verriegelung" umstellen. Dazu stellen wir den rückseitigen Codierschalter 3 auf "on" (siehe Abb. 2). Wird jetzt eine Fahrstraße gestellt, z.B. "A1", so kann danach die kreuzende Fahrstraße "A2" nicht durch einfachen Tastendruck gestellt werden. Bei Betätigung der Taste "A2" blinken beide Leuchtdioden "A1" und "A2". Erst wenn man durch einen zweiten Druck auf die Taste "A2" innerhalb von 10 Sekunden die Anforderung der Fahrstraße "A2" bestätigt, folgt die Ausführung des Stellbefehls. Andernfalls hört nach 10 Sekunden "A2" auf zu blinken und Fahrstraße "A1" bleibt gültig.
Die gültige Fahrstraße "A1" wird freigegeben durch Betätigen der Tasten "clear" und anschließend "A1". Die LED bei "A1" erlischt.
Sollen auch einprogrammierte Folgestraßen mit in die Überprüfung und Verriegelung einbezogen werden, muß der Codierschalter 4 auf "on" gestellt werden.

Externe Ansteuerung des Memory

Über die Rückmeldemodule (decoder S88) ist das Memory von der Anlage her, das heißt über Schalt- oder Kontaktgleise ansprechbar. Fahrstraßen können so angefordert und auch freigegeben werden. Über Kontaktgleise als besetzt gemeldete Gleise verhindern beim Betrieb mit eingeschalteter Verriegelung, daß die entsprechende Weichenstraße geschaltet werden kann. Beim Einsatz zur Schattenbahnhofsteuerung findet das Memory auf diese Weise automatisch das freie Gleis und führt den nächsten Zug entspechend auf das freie Abstellgleis.

An jedes Memory können bis zu 3 Decoder S88 angeschlossen werden. Sie sind in der Reihenfolge des Anschlusses vom Memory aus gesehen den Fahrstraßen A1...A8 (1.Decoder), B1...B8 und C1...C8 zugeordnet. Die Kontakte 1 bis 8 des Decoders fordern die zugehörigen Fahrstraßen an. Dagegen geben die Kontakte 9 bis 16 die Fahrstraßen wieder frei bzw. verhindern das Stellen.

Eine Codierung der Rückmeldemodule ist nicht erforderlich, da sich die Zuordnung aus der Reihenfolge des Dateneingangs am Memory automatisch ergibt (Schieberegister). Ein Beispiel:
Ein aus A kommender Zug hält vor dem Siganl 7 an. Über Kontakt "a", der an die Buchse 1

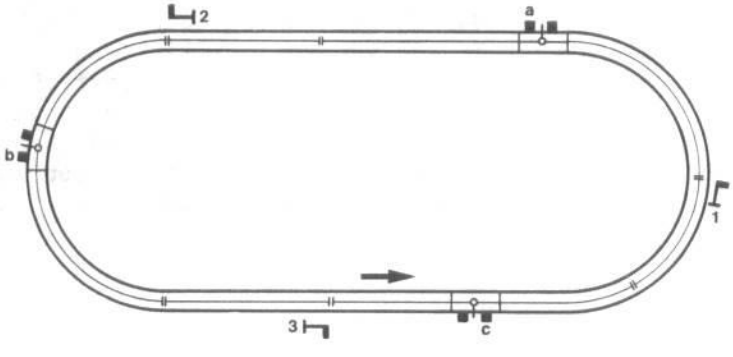
am Decoder A angschlossen ist, fordert er die Weichenstraße "A1" an. Diese wird jedoch nicht sofort geschaltet. Auf Gleis D steht nämlich noch ein Zug. Dieser blockiert über den Kontakt "d", der an die Buchse 9 am Decoder A angeschlossen ist, das Stellen der Fahrstraße "A1". Erst wenn der Zug Gleis D verlassen hat und damit Kontakt "d" freigegeben wurde, kann die Fahrstraße "A1" gestellt werden und der zweite Zug auf Gleis D einfahren.
Auf diesem Prinzip lassen sich sowohl Blockstreckenschaltungen als auch Schattenbahnhofsteuerungen aufbauen. Wichtig ist dabei, daß es sich jeweils um Dauer-Kontakte handelt, also am besten Kontaktgleise (5145 oder 2295). Schaltgleise oder Reedkontakte sind wegen ihres Moment-Kontaktes zur Gleisbesetztmeldung oder Weichenstraßen-Anforderung nicht geeignet, da die Informationen im Rückmeldemodul beim Anschluß an das Memory nicht gespeichert werden (das Rückmeldemodul wird permanent eingelesen). Beim Einsatz von Schaltgleisen sind Zusatzspeicher oder -relais erforderlich.
Wer seinen Schattenbahnhof oder seine Blockstreckenautomatik über das Memory oder über das Interface steuern möchte, muß die entsprechenden Kontaktgleisstrecken vorsehen. Für Blockstrecken und Schattenbahnhöfe ist in jedem Fall in allen Halte-Abschnitten vor Signalen eine solche Strecke zur Besetztmeldung erforderlich.
Eine eigene Kontaktstrecke für das Schließen des Signals 7 ist im vorangegangenen Beispiel nicht notwendig, da diese Schaltung als Folgeschaltung zu "A1" eingegeben werden kann und somit dann automatisch ausgeführt wird, wenn Kontakt "a" freigegeben ist.

Anwendungsbeispiele

<u>Blockstreckenbetrieb über Schaltgleise gesteuert (1-2 Züge)</u>

Der Aufbau der Anlage entspricht dem vom konventionellen Betrieb her gewohnten Schema. Durch Blocksignale am Anfang jedes Blockabschnitts gesichert, wird die Gesamtstrecke einer Fahrtrichtung in Blockabschnitte unterteilt. Die einzelnen Abschnitte besitzen etwa die Länge von 2 Zügen; jeweils in der Mitte befindet sich ein Schaltgleis.
Ein Zug, der in einen Blockabschnitt einfährt, muß das Blocksignal auf "Hp0" (rot) stellen, um die Einfahrt eines weiteren Zuges zu verhindern. Da er den vorangegangenen Blockabschnitt verlassen hat, kann er gleichzeitig dieses Signal auf "Hp1" (grün) stellen. Es lassen sich also immer 2 Schaltbefehle zusammenfassen. Bei der Memory-Programmierung ist es auf jeden Fall wichtig, die eingegebenen Schaltbefehle zu dokumentieren. Dazu liegt dem Memory - sozusagen als Handhabungsvorschlag - ein "Programmierbogen" bei.
Unterteilt ist dieser Programmierbogen in 8 Spalten für Fahrstraßen und 20 Zeilen für die 20 möglichen Schaltbefehle sowie zwei weitere Zeilen zum Eintragen der Anforderungs- und Freigabekontakte. Die Spalten besitzen ferner Felder für die Magnetartikelnummer, die Stellung und die Dauer des Schaltbefehls (D von 1 - 8). Beim Blockstreckenbetrieb mit Steuerung über Schaltgleise ist eine Verriegelung nicht erforderlich bzw. nicht sinnvoll, da sich dadurch keine zusätzliche Sicherheit ergibt.

Ringstrecke mit 3 Blockstrecken – über Schaltgleise gesteuert
(Programmierbogen 1)



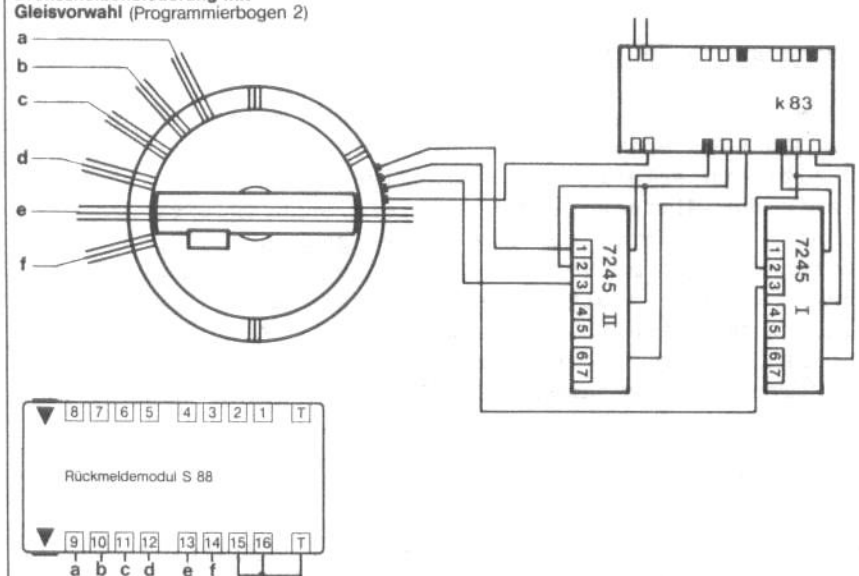
Rückmeldemodul A
Kontakt 1 – a
Kontakt 2 – b
Kontakt 3 – c

Die hier aufgeführten Beispiele sind für alle Modellbahnsysteme anwendbar. Bei den miniclub-Signalen sind lediglich die 10-V-Lämpchen gegen solche für 16 V Betriebsspannung (Art.Nr.33774) auszutauschen.

Drehscheibenschaltung mit Gleisvorwahl

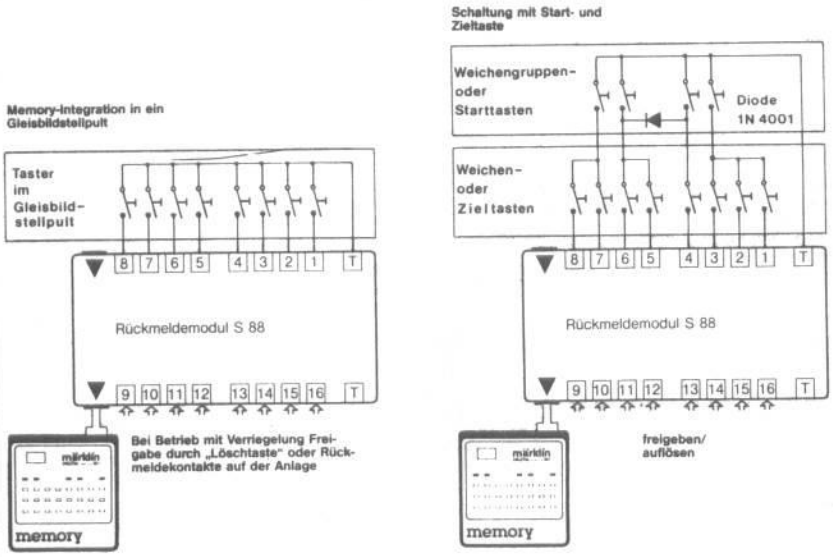
Die Ansteuerung der Drehscheibe über das Digital-System erfordert zwei Universal-Fernschalter, da die Drehscheibe nicht über Masse, sondern über den Lichtstrom gesteuert wird. Die Stromversorgung muß daher an eine gelbe Buchse des Decoders k83 angeschlossen werden. Der Universalschalter I betätigt den Rastmagneten, der auch den Motorstrom für die Drehung ein- und ausschaltet. Der Universalschalter II bestimmt die Drehrichtung. Er muß bei dieser Ansteuerung vorher eingestellt werden, falls die Drehrichtung der Bühne geändert werden soll.
"A1" bis "A6" schalten den Rastmagneten ein, bis die entsprechenden Gleiskontakte "a" bis "f" die Freigabe auslösen. Diese Gleiskontakte werden an den Mittelleiter angeschlosssen nd über je einen 560 KOhm Wiederstand mit dem Rückmeldemodul verbunden. Da das Digital H0-System Impulse mit positiver und negativer Flanke liefert, reicht der Negativanteil aus, um den Rckmeldemodul auszulösen. Dann schaltet die Folgestraße "A8" den Magneten aus. Die Bühne läuft noch solange weiter, bis die richtige Stellung zum Gleisstutzen erreicht ist. Bei "A7" erfolgt die Freigabe sofort. Dadurch läuft die Bühne ein Gleis weiter. So können auch die Zufahrtgleise angesteuert werden, die im Gegensatz zu den Schuppengleisen a-f keine Stromabschaltung besitzen. Das Erreichen dieser Position läßt sich allerdings über einen Rückmeldekontakt nicht kontrollieren.

Drehscheibensteuerung mit Gleisvorwahl (Programmierbogen 2)



Memory im Gleisbildstellwerk

Ein Fahrstraßenstellpult bietet ganz besonders in einem Gleisbildstellpult große Vorteile. Durch die Ansteuerung über das Rückmeldemodul ist eine Integration sehr einfach zu erreichen. Dabei erfolgt die Anforderung der Fahrstraßen über einfache Taster, die an die Buchsen des Rückmeldemoduls angeschlossen werden. Stellt man das Memory auf "Betrieb mit Verriegelung" ein, dann ist das Stellen kreuzender Fahrstraßen erst nach Freigabe der ersten Straße möglich. So läßt sich eine sehr hohe Betriebssicherheit zu erreichen. Die Freigabe kann durch die Züge erfolgen. Man kann aber auch "Löschtasten" vorsehen. Werden die Freigabekontakte an den Rückmeldemodulen an Kontaktstrecken auf der Anlage angeschlossen, so läßt sich das versehentliche Stellen einer Fahrstraße auf ein schon besetztes Gleis verhindern.



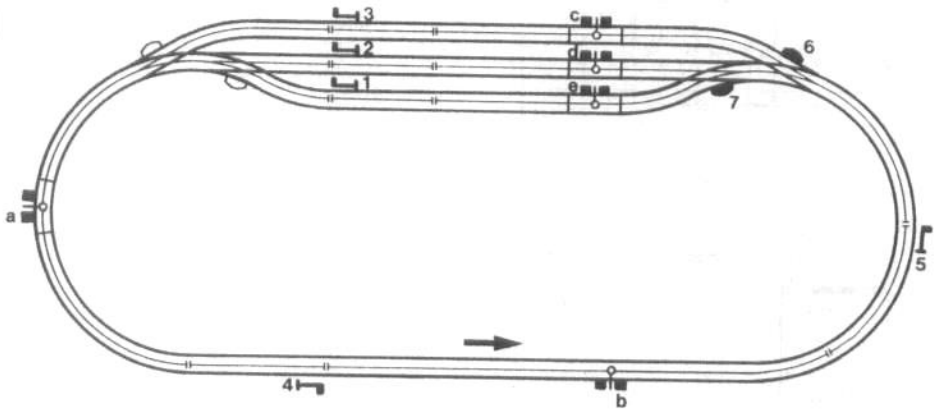
Ordnet man - schaltungstechnisch gesehen - zwei Taster hintereinander an, so kann man die bei der Bundesbahn übliche Stellung von Weichen und Signalen über Weichengruppentaste und Weichentaste bzw. Start- und Zieltaste nachbauen. Es ergeben sich hier vor allem zwei Vorteile:

1) Die Verknüpfungen sind nicht durch Kabelverbindungen festgelegt, sondern durch ein Programm, das sich sehr einfach und schnell ändern läßt.
2) Der Stromverbrauch erreicht keine hohen Spitzenwerte, die sonst bei gleichzeitigem Stellen mehrerer Weichen auftreten. Die Weichen und Signale werden automatisch nacheinander gestellt.

Blockstreckenbetrieb mit Ausweichgleis (Schattenbahnhof)

Ein Zug, der den Schattenbahnhof verläßt, fordert beim Passieren des Schaltgleises "a" die Fahrstraße "A1" an und schaltet dadurch die Ausfahrt auf "Rot". Gleichzeitig erhält der nächste Zug Einfahrt in den Schattenbahnhof "a", gibt dabei zusätzlich die Straßen "A3" bis "A5" sowie "B3" bis "B5" frei. Beim Passieren von "b" wird "A2" angefordert (Signal 4 rot). Der wieder in den Schattenbahnhof eingefahrene Zug löst je nach Gleis den Kontakt "c", "d" oder "e" aus. Der Kontakt "c" z.B. fordert die Fahrstraße "A3" an und gibt "A1" und "A2" frei. Da alle drei Gleise die gleiche Freigabe durchführen sollen, können die Kontakte "c", "d" und "e" mit Buchse 9 und 10 am Decoder verbunden werden. Nun würde dieses Zusammenschalten normalerweise auch die Anforderungen der Weichenstraßen "A3" bis "A5" miteinander verbinden. Damit das nicht geschieht, müssen sie entkoppelt werden. Der Strom von "c" darf zwar durch die Diode zu den Buchsen 9 und 10 fließen, aber nicht über die Verbindung zurück zur Buchse 4 und 5 (Kontakte "d" und "e").
Die Folgestraßen "B3" bis "B5" werden immer erst dann geschaltet, wenn "A1" freigegeben ist, da "1g", "2g" und "3g" die Eingabe "A1" kreuzen. Dadurch sind Auffahrunfälle ausgeschlossen.

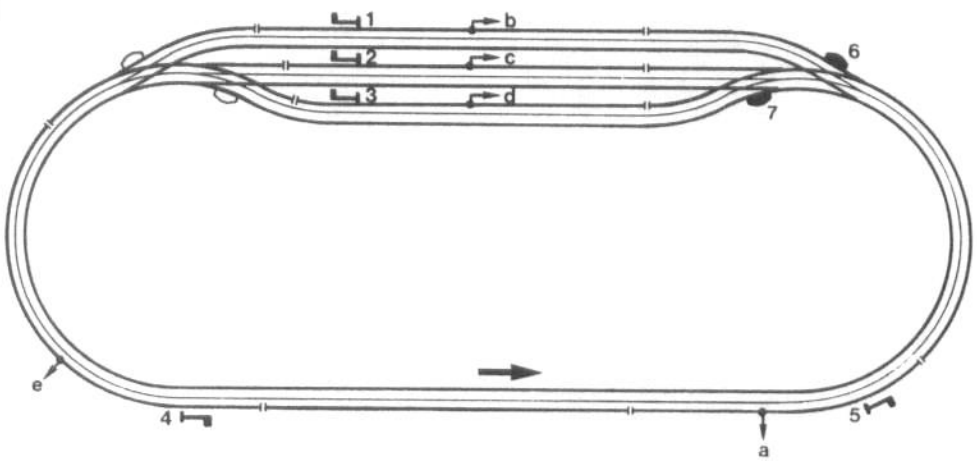
Ringstrecke mit Ausweichgleisen – über Schaltgleise gesteuert (Programmierbogen 3)



Blockstreckenbetrieb mit Schattenbahnhof

Der grundlegende Unterschied zum vorangegangenen Beispiel besteht in der unterschiedlichen Kontaktsteuerung und der daraus folgenden anderen logischen Verknüpfung der Fahrstraßen. Es wird hierbei ausgenutzt, daß der Freigabekontakt auch das Stellen einer Weichenstraße verhindern kann. Dies ist der Fall, wenn der Freigabekontakt während der Anforderung belegt ist. Auf diese Weise verhindert ein stehengebliebener Wagen das Schalten der blockierten Fahrstraße. Erst wenn dieses Hindernis beseitigt ist, läßt sich die angeforderte Fahrstraße schalten. Die gezeigte Schattenbahnhofsteuerung beruht auf diesem Prinzip. Der vor dem Einfahrsignal zum Schattenbahnhof stehende Zug fordert alle Gleise an. Jetzt wird in der Reihenfolge"A1" bis "A3" geprüft, welches Gleis frei ist. Diese Fahrstraße wird dann geschaltet. Sie enthält zuletzt den Befehl "5g", der das Einfahrsignal 5 auf "Grün" stellt. Die Folgestraßen "C1" bis "C3" stellen das Einfahrsignal wieder auf "Rot" und fordern ihrerseits als Folgestraßen "B1" bis "B3" an. Die Freigabe erfolgt hier über die Kontaktstrecke "e". Gleichzeitig wird dabei "C8" angefordert und die Ausfahrsignale des Schattenbahnhofs auf "Rot" gestellt. Läßt man die Fahrstraßen "B.." weg, so kann man durch Stellen der Ausfahrt am Schattenbahnhof von Hand eine beliebige Zugreihenfolge realisieren. Über eine Zeitsteuerung - durch einen Computer - ist so ein Fahrplanbetrieb der Züge zu realisieren. Haben die Gleise des Schattenbahnhofes nicht alle die gleiche Länge, so kann man vor dem Einfahrsignal eine Kontaktstrecke so anbringen, daß ein langer Zug sie belegt, ein kurzer Zug jedoch nicht. Dieser Kontakt kann dann - wenn notwendig über Entkopplungsdioden - das Stellen der Weichenstraßen der zu kurzen Gleise verhindern.

Blockstreckenbetrieb mit Ausweichgleisen – Steuerung durch Kontaktstrecken (Programmierbogen 4)



SPS

Speicherprogrammierbare Steuerungen sind das Kernstück der Automatisierung. Mit diesen Geräten können je nach Funktionsumfang Aufgaben wie Steuern, Regeln und Rechnen, Bedienen und Beobachten, Melden und Protokollieren ausgeführt werden. Ganz allgemein ist ein Automatisierungsgerät ein Automat mit Ein- und Ausgängen zum Anschluß an einen technischen Prozeß. Aufgrund eines Programms trifft er Entscheidungen, die je nach Eingangssignalen eine Ausgabe zur Folge haben. SPS-Systeme sind speicherprogrammierbare Steuerungen und haben die Struktur von Rechnern mit einem Zentralprozessor, einem Arbeitsspeicher, einer Ein-Ausgabelogik und einem Bus-System. Die Peripherie auf der Ein- und Ausgabeseite ist wie die Programmiersprache auf die besonderen Belange der jeweiligen Steuerungstechnik ausgerichtet. SPS-Systeme sind also anwendungsorientierte Systeme, mit denen sich relativ einfache Verknüpfungen und Ablaufsteuerungen realisieren lassen. Das Memory-Modul im Märklin Digital HO-System ist von seiner Funktion her ein SPS-System, das jedoch hinsichtlich seiner Programmierung Grenzen kennt. Wird es durch das Interface-Modul ersetzt und die Aufgaben des Memory-Moduls auf einen Personal Computer übertragen, sind alle Möglichkeiten einer programmierbaren Steuerung gegeben.

Das Programm einer SPS ist eine Folge von Anweisungen. In der DIN 19237 wird ein Programm als die Gesamtheit aller Anweisungen und Vereinbarungen für die Signalverarbeitung durch eine zu steuernde Anlage definiert.

Wichtig bei der Unterscheidung der verschiedenen Steuer- und Regelungstechnik ist sich darüber klaren zu sein, daß Steuern ein Verfahren ist, um einen Prozeß planmäßig zu beeinflussen. Die DIN-Normen definieren Steuerung und Steuerung des Ablaufs in einem System als die Reaktion eines Systems auf mehrere Eingangsggrößen, die sich auf Ausgangsgrößen auswirken. Kennzeichnend für das Steuern ist der offene Wirkungskreis über die einzelnen Übertragungsglieder oder die Steuerstrecke. Eine Steuerung liegt also vor, wenn Eingangsgrößen nach einer festgelegten Gesetzmäßigkeit Ausgangsgrößen beeinflussen. Die Auswirkung einer Störgröße wird nicht ausgeglichen. Diese Verhältnisse liegen beim Einsatz des Memory-Moduls vor. Über den Rückmeldemodul können Eingangssignale erfaßt und Reaktionen ausgelöst werden.

Regeln ist im Gegensatz zum reinen Steuern ein Vorgang, bei dem die zu regelnde Größe fortlaufend erfaßt mit einer vorgegebenen Zielgröße der Führungsgröße verglichen und abhängig vom Ergebnis dieses Vergleichs im Sinne einer Angleichung an die Führungsgröße beeinflußt wird. Kennzeichnend ist ein geschlossenes System, der Regelkreis. Der Vorgang der Regelung wird auch dann als kontinuierlich angesehen, wenn die Regelgröße durch hinreichend häufige Stichprobenentnahme erfaßt wird. Dies ist bei der Abfrage der Rückmeldemodule gegeben. Eine wesentliche Aufgabe des Regelns ist die Beseitigung des Einflußes von Störgrößen. Ein Beispiel für eine Regelung, die sich mit dem Memory-Modul allein nicht erreichen läßt, ist das Hintereinanderherfahren von Lokomotiven oder Zügen, die aufgrund ihrer Konstruktion bei derselben angewählten Fahrstufe eine unterschiedliche Geschwindigkeit erreichen. Wird der Regelkreis, beispielsweise das Schienenoval mit den entsprechenden Sensoren ausgestattet, regelmäßig abgefragt, so fahren die Lokomotiven oder Züge hintereinander her, wobei jede Lokomotive mit einer unterschiedlichen Fahrstufe angesteuert wird. Über die Sensoren wird festgestellt, ob die Lokomotiven dabei sind, aufeinander auf zu fahren oder zu weit von einander abfallen aufgrund unterschiedlicher Fahrgeschwindigkeiten und die Differenz durch die Wahl der geeigneten Fahrstufe ausgeglichen.

Der Aufbau einer SPS umfaßt alle Funktionsgruppen einer Informationsverarbeitung. Es gibt Eingabe/Ausgabe sowie ein Rechenwerk, den Prozessor, mit dem integrierten Programmspeicher. Beim Memory-Modul von Digital HO wird der Programmspeicher über Tastendrucke vom Keyboard gefüllt. Wichtig im Zusammenhang mit dem Memory-Modul ist, daß nur Informationen gespeichert werden können, die über das Keyboard eingegeben werden und keine Informationen vom Steuergerät Control 80. Die zu steuernde Anlage, in unseren Fall die Modellbahnanlage liefert über Sensoren also die Schaltgleisstücke Eingabesignale an die Eingabeeinheit, den Rückmeldedecoder des Memory-Moduls. Diese Signale werden fortlaufend durch das im Programmspeicher des Memory-Moduls hinterlegte Steuerprogramm verarbeitet. Das Ergebnis der Verarbeitung wird über die Ausgabeeinheit, also die Ansteuerung der Weichen und Signale ausgegeben.

Das Memory-Modul arbeitet zur Dateneingabe und zur Datenausgabe mit digitalen Signalen. Grundlage des Steuergeschehens beim Digital HO-System sind binäre Signale, die nur zwei Werte annehmen können. Mögliche Werte können z.B. sein,
Zustand 1, Signal auf rot - Weiche rechts,
Zustand 0 - Signal grün bzw. Weiche gerade.
Ein Wechsel des Signalzustands einer Binärzahl stellt einen Wechsel zwischen zwei möglichen Werten dar, also Umschalten des Signals von rot auf grün oder der Weiche von gerade auf abbiegend. Ein Bit ist damit die kleinste mögliche Informationseinheit.

Mehrere binäre Signale zusammengefaßt ergeben nach einer bestimmten Zuordnung (Code) ein digitales Signal. Die Information vom Rückmeldemodul, beispielsweise faßt Bits zu Bytes zusammen, die dann ausgewertet werden müssen. Genau gleich funktioniert die Informationsübermittlung an die Decoder, die Weichen und Signale schalten, in Worten, die wiederum binäre Informationen zusammenfassen. Im Gegensatz zu Computersystemen, die entweder mit 8 oder 16 Bit Datenbreite arbeiten, arbeitet das Märklin Digital HO in unterschiedlichen Wortbreiten, die je nachdem, ob Magnetartikel oder Lokomotivmotoren angesteuert werden.

Das Memory-Modul kann nur Methoden der linearen Programmierung umsetzen. Anweisungen werden linear, das heißt in einer festen Reihenfolge abgearbeitet, in der sie auch im Programmspeicher hinterlegt sind. Dies bedeutet: erst wird eine Reihe von Weichen gestellt, um die Fahrstraße festzulegen, anschließend wird das Signal freigegeben. Beeinflußt wird der lineare Ablauf des unter einer Fahrstraßentaste gespeicherten Programms also die verschiedenen Schritte, mit denen Magnetartikel gesteuert werden, lediglich durch das Rückmeldemodul, mit dem die Fahrstraße aufgerufen und anschließend wieder freigegeben werden kann. Die lineare Programmabarbeitung ist meist nur für einfache, nicht so umfamgreiche Steuerungsprogramme verwendbar. Deshalb kann das Märklin Digital HO Memory-Modul auch nur 20 Schritte unter einem "Programmnamen" also einer Fahrstraße verwalten. Komplexere Entscheidungen sind natürlich problemlos möglich, nur ist dazu das Interface-Modul und ein Computer, der dann den Programmablauf überwacht und auch strukturierte Programmverarbeitung ermöglicht, notwendig.

Das Memory-Modul arbeitet nur mit einfachen Grundverknüpfungen. Eine Fahrstraße ist aufgerufen oder ist nicht aufgerufen, eine Fahrstraße ist freigegeben oder nicht freigegeben. Grundsätzlich lassen sich speicherprogrammierbare Steuerungen mit den drei logischen Grundelementen 'nicht', 'und' und 'oder'bearbeiten. Diese Funktionen stehen mit dem Interface Modul zur Verfügung, das dann die jeweiligen Möglichkeiten der dort eingesetzten Programmiersprache ausnutzt. Um speicherprogrammierbare Steuerungen mit logischen Verknüpfungen zu realisieren, reichen diese 3 Basisgrundfunktionen aus. Für den praktischen Einsatz auf den Schienensträngen muß genaugenommen die logische Verknüpfungskette modifiziert werden. Die logische Negation 'nicht', die Grundverknüpfung 'und' und die Grundverknüpfung 'exklusiv oder', also entweder oder, reichen aus, um alle in der Praxis auftretenden Probleme zu lösen. Das einfache 'oder', das sowohl den einen als auch den anderen Ausgangszustand zuläßt, wird kaum, höchstens in Ausnahmefällen einsetzbar sein, wenn Kollissionen der Züge verhindert werden sollen.

Einführung in die Regelungstechnik

Basis einer Regelung, die sowohl mit einer Stellgröße als auch mit einer Regelgröße, also einer Eingangs- und Ausgangsgröße arbeitet, ist der Begriff der Regelstrecke. Beispiele hierfür sind Elektromotoren, deren Drehzahl geregelt werden soll, beispielsweise der Lokomotivmotor, für den insgesamt 15 verschiedene Fahrstufen also Drehzahlgeschwindigkeiten festgelegt werden können oder der Kurs, den der Zug auf einer Anlage fahren soll, also die Verküpfung der unterschiedlichen Fahrstraßen.

Die komplexen Funktionen, die unter Umständen hinter dem einfachen Begriff Regelstrecke stehen, werden symbolisch durch einen rechteckigen Block dargestellt. Dieser Block hat eine Eingangsgröße, die Stellgröße y und eine Ausgangsgröße, die als Regelgröße x bezeichnet wird. Am Beispiel der Digital HO-Anlage ist die Eingangsgröße y ein Rückmeldekontakt, der über das Rückmeldemodul ausgelöst wird, und die Stellgröße y, die Fahrstraße, die freigegeben oder der Lokomotivmotor, der mit einer bestimmten Drehzahl gestartet wird. Ein Beispiel für einen geschlossenen Regelkreis, der auch die Funktionen

Soll-Ist-Vergleich umfaßt, ist das Hintereinanderfahren von Lokomotiven, die aufgrund ihrer Konstruktion, also der Motoren und Getriebe, bei gleicher Fahrstufenwahl eine unterschiedliche Geschwindigkeit erreichen. Die Regelaufgabe besteht darin, die zwei Lokomotiven mit unterschiedlicher Fahrstufensteuerung so hintereinanderherfahren zu lassen, daß sie an den entsprechenden Blockabschnitten kaum - nach Möglichkeit nicht - abgebremst werden müssen. Da sich das Problem der Regelstrecke, also die Steuerung der langsam fahrenden Lok, mit der entsprechenden Auswahl der Fahrstufe für die schneller fahrende Lok nicht ohne weiteres alleine einstellen läßt, ist es zweckmäßig, dieses Problem durch eine Regeleinrichtung zu lösen. Im Grundsatz handelt es sich um einen Kreislauf. Die Regeleinrichtung hat ein Eingangssignal, die Regelgröße x und das Ausgangssignal, die Stellgröße y. Die Eingangssignale werden durch die Signale des Rückmeldemoduls, das Schaltgleise überwacht, geliefert und die Stellgröße ist die Wahl der geeigneten Fahrstufe. Der Regelkreis besteht aus einer Regelstrecke, dem Gleisoval mit den Rückmeldemodulen und der Regeleinrichtung, dem Computerprogramm, das dafür sorgt, daß die beiden Fahrstufen der Lokomotiven, die hintereinanderherfahren, aufeinander abgestimmt sind. Das besondere Merkmal eines Regelkreises ist der geschlossene Wirkungsweg. Störgrößen beeinflussen solche Regelkreise. Störgrößen können sein, wenn eine Lokomotive entgleist und damit nicht weiterfährt oder wenn bei einer bestimmten Fahrstufe, bedingt durch den Widerstand, den Weichenstraßen erzeugen, die Geschwindigkeit abfällt.

Diese Regelkreise unterliegen verschiedenen Eigenschaften der Regelstrecke. Die Eigenschaften einer Regelstrecke, die unter dem Begriff Zeitverhalten zusammengefaßt werden, stellen sich dar, als beispielsweise die Übergangsfunktion. Die Übergangsfunktion wird in einer Regelstrecke durch die sprunghafte Veränderung der Stellgröße y ausgelöst. Die Regelgröße, also das Ausgangssignal, folgt der plötzlichen Information nicht unmittelbar. Das bedeutet in unserem Beispiel, wenn die Lokomotive, die aufgrund ihrer physikalischen Konstruktion schneller läuft als eine andere hinter der langsameren Lokomotive herfährt, werden diese unterschiedliche Verhaltensweisen der Lokomotiven zueinander über die Schaltgleise erfaßt. Die Schaltgleisimpulse werden ausgewertet und entsprechend des Ergebnisses der Auswertung wird die Geschwindigkeitsstufe für die nachfolgende Lokomotive umgesteuert. Im Idealfall besteht das Gleisoval aus einer Folge von Rückmeldemodulen, die so beispielsweise auf eine halbe Gleislänge genau, was bei mittlerer Fahrgeschwindigkeit einer Lokomotive einer Impulsfolge unter einer Sekunde entspricht, eine relativ feinfühlige Regelung des Regelkreises ermöglicht. In der Praxis werden jedoch zwischen den einzelnen Schaltkontakten andere Gleisstücke eingefügt, so daß das Computersystem, das die Fahrt der Züge verfolgt, die Impulse in größeren Zeitabständen erhält. Aus Sicherheitsgründen empfiehlt es sich, zwischen den beiden Lokomotiven nicht nur den Abstand eines Schaltkontaktes sondern den Abstand von mindestens zwei Schaltkontakten einzuhalten. Damit erfolgt die Regelung der hintereinanderherfahrenden Lokomotiven in unserem Beispiel sind sie durch den Abstand von mindestens zwei Schaltrückkontakte, die Rückmeldemodule informieren, getrennt und dort wiederum durch mehrere Gleisstücke auseinandergezogen in diskreten Zeiteinheiten, die wesentlich größer sind als für eine kontinuierliche Steuerung notwendig.
Bei der Steuerung der Lokomotivfunktionen bedeutet dies, daß die Regeleinrichtung, also das Computersystem, das seine Informationen über die Rückmeldekontakte erhält, mit einigem Zeitverzug feststellen kann, ob die Lokomotive abfällt oder aufholt. Die verfolgende Lokomotive kann also beim ersten Start des Regelkreises nur relativ ruckartig beschleunigt oder abgebremst werden. Erst wenn die Lokomotiven den Regelkreis einige Male durchfahren haben, ist der Regelkreis, der dem Schwingungsverhalten folgt, einigermaßen ausgeglichen, so daß die Lokomotiven ohne große Beschleunigungsänderungen hintereinanderherfahren. Kennzeichnend für die Stabilität unseres Regelbeispiels zwei hintereinanderherfahrender physikalisch verschieden schnell laufender Lokomotiven ist die Regelabweichung oder die Regeldifferenz. Die Regelabweichung beträgt mindestens eine Fahrstufe und ergibt sich auf beiden Seiten eines diskreten Schaltvorgangs betrachtet, als Differenz von mindestens zwei Fahrstufen zwischen der idealen Fahrgeschwindigkeit der Lokomotive. In der Steuerungspraxis bedeutet dies, daß die verfolgende Lokomotive in einem Spielraum, der mindestens 3 Fahrstufen umfaßt, geregelt werden muß, um der vorausfahrenden Lokomotive in einem festgelegten Abstand folgen zu können.

Die Vorteile des Digital-Betriebs

Der unabhängige Mehrzugbetrieb von maximal 80 Triebfahrzeugen lohnt sich natürlich vorrangig an Verkehrsknotenpunkten wie Bahnhöfen, Verladeeinrichtungen, Bws, Abstellgruppen usw. Vollkommen freizügiger Verkehr von Rangierloks, Triebfahrzeugwechsel ohne besonderen Aufwand mit Gleistrennstellen, problemloser lokspezifischer Fahrtrichtungswechsel und die bequeme Bedienung der Märklin-TELEX-Kupplung sind nur einige der vielen Vorzüge.

Die Bausteine für eine kombinierten Anlage

Als Stromversorgung benötigen Sie zunächst einen Transformator. Zum Märklin-Digital-HO-System gehört der "transformer" 6002. Er leistet 52 VA. Eventuell läßt sich aber auch ein vorhandener Lichttrafo 6611 einsetzen. Dies schränkt allerdings die Leistungsfähigkeit der angeschlossenen Zentralelektronik etwas ein. Der Transformer liefert im Gegensatz um Lichtransformator 16 Volt 3 Ampere.

Zweiter Baustein ist die Zentraleinheit "central unit" 6020. An dieses "Herz" jeder Digitalanlage sind alle weiteren Systemelemente anzuschließen.

Das Fahrpult "Control 80" 6035 könnte gewissermaßen das "Etappenziel" einer ersten Ausbaustufe darstellen. Zusammen mit Loks der Serie 3600 erlaubt diese Gerätekombination schon den unabhängigen Mehrzugbetrieb. Wer sich das Märklin-Digital-HO-System erst später anschaffen will, sollte dennoch bei einem eventuellen Triebfahrzeug-Neukauf gleich Loks mit Digital-Empfangsmodulen wählen. Diese Fahrzeuge unterliegen auch auf konventionell versorgten Anlagen keinerlei Fahrbeschränkungen. Lediglich die jeweilige Zusatzfunktion (z.B. Lokbeleuchtung oder Telex-Kupplung) läßt sich mit den normalen Transformatoren nicht einschalten.

Beim Umstieg auf die Digital-Technik um, so stehen ihm in diesem Fall schon Loks mit Empfangsmodulen zur Verfügung. Durch den Nachrüstmodule 6080 lassen sich vorhandene Fahrzeuge umrüsten. De Umbau benimmt der Fachhandel. Mit einem anderen Mdl, dem 6081 können auch Lokomtien mit Gleichstrommotor umgeüstet werden.

Die digitale Ansteuerung von Magnetartikeln erfordert neben diesen Bausteinen ferner das Magnetartikel-Stellpult "Keyboard" 6040 sowie die Magnetartikel-Empfangsmodule "Decoder k 83" 6083. An einer beliebigen Stelle des Digital-Gleissystems angeschlossen, decodiert jedes dieser Module die Stellbefehle für jeweils 4 zweispulige (oder 8 einspulige) Magnetartikel.

Vorbereitungen zum Einsatz von Märklin-Digital-HO

Als Beispiel dient die Teilumrüstung eine mittelgroße Märklin-Anlage voraus, die sowohl durchgehende Streckengleise als auch einen elektrisch abtrennbaren Rangierbereich besitzt. Hierbei ist es gleichgültig, ob es sich um ein Bw, verzweigte Industriegleisanlagen, Abstellgruppen oder ähnliches handelt.
Wichtig ist nur die klare elektrische Trennungsmöglichkeit zwischen selbständig funktionsfähigen Bereichen. Meist werden solche Anlagenabschnitte ohnehin von separaten Fahrgeräten versorgt. In diesem Fall sind lediglich die Ausgangskontakte der Zentralelektronik (0 = Masse und B = Bahnstrom) mit dem Anschlußgleis des betreffenden Anlagenabschnitts zu verbinden.
Die Funktion von Trenngleisabschnitten an Signalen, von Abstellgruppen usw. bleibt auch in Digitalstromkreisen voll erhalten. Allerdings ist parallel zu sämtlichen Fahrstromtrennstellen auf der gesamten Anlage jeweils ein Widerstand (1,5 kOhm/1/4 Watt) einzubauen, damit die Speicher der Digital-Lokmodule die letzten Informationen von der Zentralelektronik nicht verlieren, sobald die Stromzufuhr länger als zwei Minuten ausbleibt. Auf konventionellen Anlagen schon vorsorglich eingebaut, stören diese Bauteile nicht, ersparen aber spätere Umrüstarbeiten.
Bei abschaltbaren Gleisbereichen weiterhin analog versorgter Anlagenteile müssen die elektrisch abgetrennten Gleise über einen 1,5 kOhm/1/4 W Widerstand allerdings vom Lichtausgang eines Trafos gestützt werden, da ein normal versorgter Fahrstromkreis nur bei aufgedrehtem Stellknopf Spannung führt.

Haben wir diese Vorbereitungen getroffen, die Zentralelektronik an einen Transformator 6002 angeschlossen und das Fahrpult 6035 rechts an der "central unit" angesteckt, steht einem ersten Betrieb nichts mehr im Wege.
Die Empfangsmodule der werksseitig ausgerüsteten Digital-Loks sind bereits codiert, also mit einer Codenummer zwischen 01 und 80 ansprechbar. Daher entfallen weitere Maßnahmen an den Triebfahrzeugen. Weden mehere Fahrzeuge desselben Typs eingesetzt müssen die Adressen jedoch geändert werden.
Modellbahner, die jedoch eine Codierung nach eigenen Vorstellungen wünschen, können die Lokmodule jederzeit einer anderen Adresse zuordnen.

Fahrt frei mit Märklin Digital HO

Herkömmliche Triebfahrzeuge fahren auf Digital-Fahrstrom-Strecken konstant mit mittlerer Geschwindigkeit. Die Züge lassen sich auf diesen Anlagenteilen aber problemlos durch Signalsysteme beeinflussen. Solche Einrichtungen sind ohnehin schon meist in Modell-Bahnhofsanlagen vorhanden.

Konstante Zugbeleuchtung

Das Märklin-Digital-HO-System erfüllt gleichzeitig den Wunsch nach konstanter Zugbeleuchtung. An den Schienen liegt stets Spannung an. Diese erlaubt nicht nur den Anschluß von Beleuchtungsartikeln und Funktionsmodellen an beliebiger Stelle der digital versorgten Gleisanlagen, sondern aus den Schienen läßt sich selbst bei Fahrzeugstillstand der notwendige Strom für Schluß-, Spitzen- und Wageninnenbeleuchtungen entnehmen.

Fahrtrichtungswechsel in Digital-Abschnitten

Einige Überlegungen verlangt der Fahrtrichtungswechsel konventioneller Wechselstromloks auf digital versorgten Kopfbahnhofsgleisen. Anhalten lassen sich "Normalloks im Digitalland" durch Fahrstromabschaltung auf Trenngleisstellen problemlos. Soll das Triebfahrzeug aber in der Gegenrichtung weiterfahren, etwa beim Wendezugverkehr oder zum Verlassen eines Abstellgleises, so müssen wir auf den betreffenden Gleisabschnitt einen Überspannungsimpuls legen. Sinnvollerweise wählt man für diesen Zweck einen Taster (1 x um), der als einfachste Lösung die Trennabschnitt-Fahrstromversorgung auf einen normalen Fahrtrafo legt. Der Fahrtrichtungswechsel läßt sich eleganter mit einer einfachen Spannungsverdopplerschaltung realisieren. Sie liefert die erforderliche Umschaltspannung und wird vom Lichtausgang eines beliebigen Märklin-Trafos versorgt. Bei dieser Lösung genügt für die Umschaltung lediglich ein kurzer Druck auf den Taster. Auch für mehrere Trenngleisabschnitte, die mit einem Überspannungsimpuls beaufschlagt werden sollen, genügt eine einzige Verdopplerschaltung. Allerdings ist jeweils ein separater Taster vorzusehen.

Analog-Digital-Trennstellen

Wenn Loks über Trennstellen zwischen den beiden Fahrstromsystemen fahren, können Schleifer die Mittelleitertrennung überbrücken. In diesem Fall bekämen die im Digital-Stromkreis stehenden Digital-Triebfahrzeuge über den "durchgerutschten" Wechselstrom den Befehl, auf Analog umzuschalten. Als Folge würden sich diese Lokomotiven ungewollt kurz in Bewegung setzen. Diese Erscheinung läßt sich aber durch die Trennstellenwippe 38512 vermeiden. Jeweils über einen Punktkontakt an den Systemtrennstellen gesetzt, heben diese Plastikteile den Schleifer einseitig an. Exakt in der Mitte kippt der Schleifschuh von einer Seite auf die andere und gewährleistet so die eindeutige Stromabnahme aus jeweils nur einem Stromkreis.
Reicht für größere Anlagen die Leistung von 52 VA der Zentralelektronik nicht aus, dann läßt sich mit einem fünfpoligen Flachkabel an der "central unit" ein Leistungsverstärker 6015 "booster" anschließen. Dieser versorgt einen weiteren Digitalabschnitt mit der Leistung des an den Verstärker angeschlossenen Transformators. Zwischen verschiedenen Digital-Gleisbereichen genügt eine normale Trennstelle ohne Schleiferwippe.

Das Märklin-Interface 6050

Das Interface bildet die Brücke zwischen dem Märklin-Digital-Steuercomputersystem und einem Homecomputer oder einem PC, der Regel- und Kontrollfunktionen übernimmt.

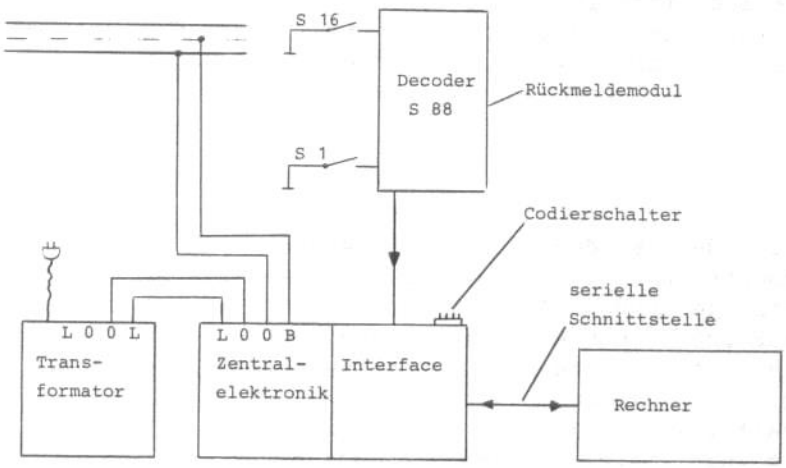
Den Sinn einer computergestützten Modellbahnsteuerung zeigen schon wenige Beispiele sehr deutlich: Schattenbahnhöfe sind in der Regel schlecht einsehbar und schwer zugänglich. Was dort abläuft, interessiert den Modellbahner während des Spielbetriebs wenig, da auch bei konventioneller Steuerung dieser Bereich meist automatisiert ist. Mit konventioneller Elektronik lassen sich diese Abläufe zwar auch bisher schon, allerdings meist aufwendig, steuern. Die Ergebnisse werden jedoch nicht immer befriedigen. Ein weiteres Beispiel: Drehscheiben und Schiebebühnen erfordern beim Betrieb ständig Aufmerksamkeit. Da diese Vorgänge recht langsam ablaufen, bewegt sich währenddessen auf anderen Anlagenteilen nicht viel. Auch hier wäre eine Automatik sinnvoll.

Gleisbildstellpulte sind relativ teuer und aufwendig in der Herstellung. Sie liefern einen guten Überblick über den aktuellen Betriebszustand auf einer Modellbahnanlage. Jedoch weicht selbst die Bundesbahn seit neustem von den althergebrachten Gleisbildstellpulten ab und setzt Bildschirme zur Anzeige ein.

Dies sind nur einzelne Punkte, zu denen sich die meisten Anwender Verbesserungen wünschen. Aber bei großen Anlagen kommt noch ein wichtiger Punkt hinzu: Meist benötigen sie mehrere Betreuer, und ein eindrucksvoller Ablauf läßt sich nur dann erzielen, wenn alle Stationen besetzt sind. Dies führt teilweise zu Zeitproblemen: Nicht immer finden alle Spielpartner gemeinsam Zeit zu Modellbahn-Aktivitäten. Wer hat sich in einer solchen Situation nicht schon einmal einen Roboter als Helfer gewünscht?

Eingaben an den Bedienelementen fließen über die Elektronik zur Modellbahnanlage. Die darauf bedindlichen Magnetartikel und Triebfahrzeuge führen die Anweisungen aus. Dazu werden über die Rckmeldemodule Informationen über den Fahrzeugstand erfaßt.

Der Computer kann über das Interface die Funktionen des Steuergeräts Controll 80, des Keyboards zur Steuerung der Magnetartikel und des Memory übenehmen. Dau kommt, daß der Computer wesentlich differzierte Ablaufsteuereungen erlaub als das Memory. Konsequent eingesetzt braucht man für Märklin Digital Modellbahnsteuerungen nur die Zentraleinheit, das Interface und einen Home- oder Personal-Computer.



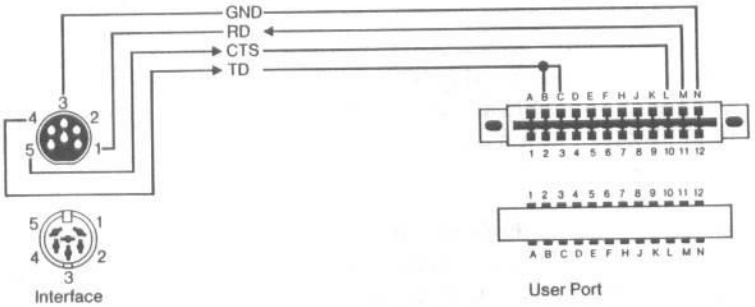
Es gibt - unterschiedliche - Computer. Da sich Märklin nicht auf einen bestimmten Computertyp festlegen wollte, wurde ein Interface entwickelt, das über eine universelle serielle Schnittstelle verfügt, die sich an praktisch alle Systeme anpassen läßt.

Die Schnittstelle und ihre Initialisierung

Wir haben die digitalgesteuerte Modellbahnanlage über das Interface mit dem Computer verbunden und wollen zunächst Keyboard und Control 80 simulieren.

Neben der Belegung der Schnittstellenkabel, die die Tabelle zeigt muß die Schnittstelle initialisiert werden.

Interface		User Port	
Pin	Leitung	Pin	Leitung
1	RD	M	PA 2
2	–	–	–
3	GND	N	GND
4	TD	B	FLAG 2
		C	PB 0
5	CTS	L	PB 7



Achtung:
Anschlußbezeichnungen genau beachten!

Für den C64 gilt:

```
OPEN2,2,0,CHR$(10)+CHR$(0)
```

Bei Geräten ab Baujahr 1986:

```
OPEN2,2,0,CHR$(138)+CHR$(0)
```

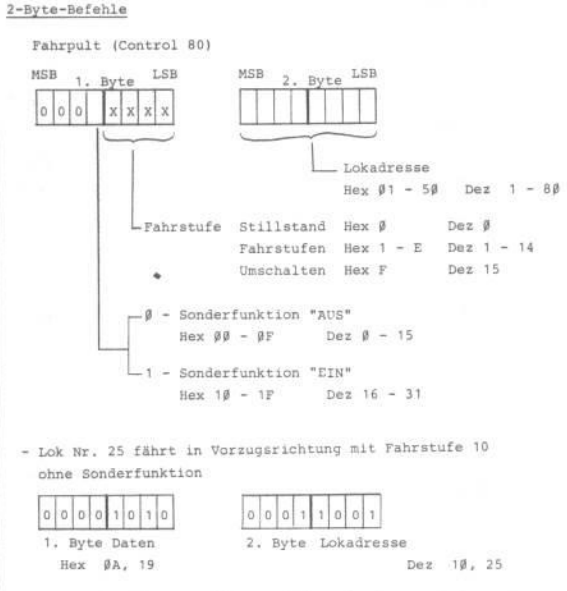
Für die verschiedenen Computersysteme ist die Schnittstelleninitiliaerung ebenfalls tabellarisch zusammengefaßt.

Wenn mit einem BASIC-Interpreter und im Direktmodus die ersten Tests ausgeführt werden bruacht man lediglich noch die Befehlszeile:

```
PRINT#2,CHR$(X);CHR$(Y);
```

Übergebe an das Interface über den Kanal 2 die Information X und die Information Y.

Jetzt brauchen Sie nur noch zu wissen, welche Informationen bei X und Y einzufügen sind und schon können Sie all das tun, was Sie auch über das Keyboard bzw. ein Control 80 erledigen können. Bevor wir zu den Werten von X und Y kommen, noch einige kleine Vorbemerkungen über das Keyboard und das Control 80. Die Funktionsweise des Keyboards ist klar, da es den konventionellen Weichenstellpulten sehr ähnelt. Für die einzelnen Weichen und Signale sind jeweils rote und grüne Tasten vorgesehen, um sie in die entsprechende Stellung zu bringen. Man muß also in den Computer nur eingeben: "Drücke rote Taste" oder "Drücke grüne Taste" und natürlich die Weichennummer.



- Lok Nr. 25 fährt in Vorzugsrichtung mit Fahrstufe 10 ohne Sonderfunktion

Steuerinformation für Lokomotiven

```
PRINT#2,CHR$(X);CHR$(Y);
```

Ähnliches gilt auch für das Control 80, besonders für die Tasten STOP/GO und FUNCTION/OFF sowie die Loknummer. Nur bei der Geschwindigkeitssteuerung sieht es etwas anders aus.

Zunächst jedoch noch einmal zu den Werten von X und Y. Statt X und Y können Zahlen von 0 bis 255 eingesetzt werden. Dies erklärt, warum bis zu 256 Magnetartikel mit Märklin Digital H0 schaltbar sind. Mit den Loknummern hat es eine andere Bewandtnis. Da die Lokdecoder intern anders aufgebaut sind, lassen sie nur Nummern zwischen 1 und 80 zu. Die jeweilige Lok- oder Magnetartikelnummer geben wir statt Y in unserem zweiten Schlüsselwort ein. Wollen wir also die Lokomotive 5 oder den Magnetartikel an der Decoderadresse 5 ansteuern, so wäre einzugeben:

```
PRINT#2,CHR$(X);CHR$(5);
```

Wie Sie vermuten, wird dagegen über unseren Platzhalter X entschieden, ob wir eine Lok oder einen Magnetartikel ansteuern. Aber damit noch nicht genug: Auch die Geschwindigkeit der Lok, der Einschaltzustand der Zusatzfunktion, die Weichenstellung (Simulation der roten oder grünen Taste) und später auch das Einlesen der Rückmeldemodule wird über den ersten Wert in unserem zweiten Schlüsselwort erledigt.

Kommen wir zurück zum Prinzip der Geschwindigkeitssteuerung beim Control 80. Anders als bei den konventionellen Transformatoren, bei denen die Geschwindigkeit annähernd stufenlos gesteuert wird (der Stellknopf beeinflußt lediglich den Abgreifschleifer auf der offenen Trafo-Sekundärspule), liegt beim Control 80 eine stufenweise Steuerung vor. Man kann den einzelnen Geschwindigkeiten auch Fahrstufen zuordnen, d.h. 14 feststehende unterschiedliche Geschwindigkeiten entsprechen den 14 Fahrstufen.

Die "Fahrstufe" 15 steuert den Richtungswechsel.
Wollen wir also z.B. die Lok Nummer 05 mit Fahrstufe 10 fahren, so geben wir in unseren Computer ein:

```
PRINT#2,CHR$(10);CHR$(5);
```

Für die Rückstellung der Geschwindigkeit auf Stillstand wäre dementsprechend einzugeben:

```
PRINT#2,CHR$(0);CHR$(5);
```

Die Zusatzfunktion

Was ist jetzt mit der Zusatzfunktion? Sinnvoll, aber zunächst etwas ungewohnt, können wir der Zusatzfunktion die Fahrstufe 16 zuordnen. Wenn Sie eingeben

```
PRINT#2,CHR$(16);CHR$(5);
```

wird in der Lok 5 die Beleuchtung (sofern stattdessen nicht Rauchgenerator oder eine Telexkupplung angeschlossen sind) eingeschaltet; die Lok bleibt also stehen.
Was wäre dann mit Fahrstufe 17? Die Lok fährt mit Fahrstufe 1 und eingeschaltetem Licht. In diesem Fall steuern wir die Lok durch die Fahrstufen 16-31. Dies ist recht einfach zu rechnen: Zählen Sie der gewünschten Fahrstufe lediglich 16 hinzu.
Mit unserem Schlüsselwort PRINT#2,CHR$(X);CHR(Y); übergeben wir also dem Rechner die Information für eine gewünschte Tätigkeit, wobei die zweite Zahl (Y) die Nummer und die erste Zahl (X) die eigentliche Steuerinformation beinhaltet.

Ansteuerung von Magnetartikeln (Weichen, Signalen usw.)
In ähnlicher Weise erfolgt auch die Steuerung der Magnetartikel. Dazu benutzen wir die gleiche Steueranweisung:

```
PRINT#2,CHR$(X);CHR$(Y);
```

Der erste Teil (mit dem "Platzhalter" X) gibt an, in welche Richtung die Weichenzunge umgelegt werden soll, der zweite Teil (Y) ist die Nummer der aufzurufenden Weiche.
Anweisung: Weiche nach links umlegen!

```
PRINT#2,CHR$(33);CHR$(Y);
```

Anweisung: Weiche nach rechts umlegen!

```
PRINT#2,CHR$(34);CHR$(Y);
```

Wie wir wissen, benötigen Spulen-Antriebe nur kurze Impulse, Demgegenüber behält ein Lokdecoder den einmal gegebenen Befehl im "Gedächtnis". Daher ist zum Ende der Magnetartikel-Eingabe(n) ein Löschbefehl erforderlich, weil sonst Dauerimpuls gegeben wird.
Anweisung: Strom für Weichenspule abschalten!

```
PRINT#2,CHR$(32);CHR$(Y);
```

In diesem Fall sollte ausnahmsweise der zweite Teil des Befehls wegfallen, so daß stehenbleibt:

```
PRINT#2,CHR$(32);
```

Wenn mehrere Stelleingaben aufeinander folgen, ist lediglich nach der letzten Stellanweisung ein Löschbefehl erforderlich, da eine Neueingabe das zuvor gegebene Kommando automatisch löscht.

Zur Numerierung der Weichen bedarf es einer besonderen Erklärung:
In den zweiten Teil der Anweisung können die Weichennummern von 0-255 eingesetzt werden.

Beispiel:
Anweisung: Weiche 128 nach links!

```
PRINT#2,CHR$(33);CHR$(128);
```

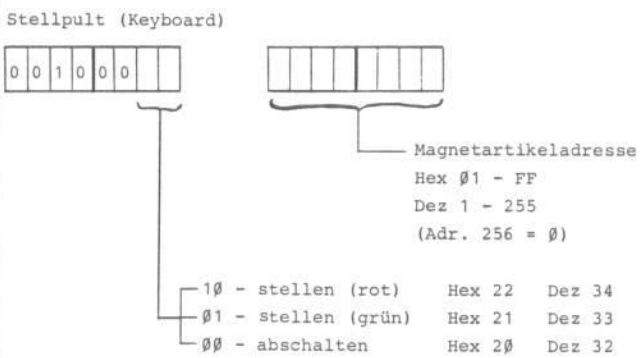
Stopbefehl nicht vergessen:

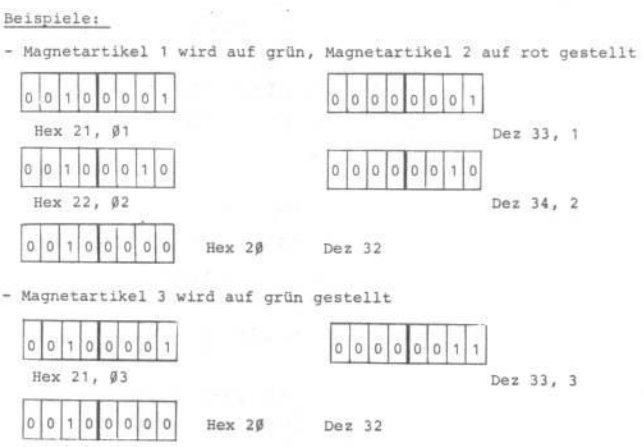
```
PRINT#2,CHR$(32);
```

Wenn Sie aufmerksam die Anleitung zum Keyboard gelesen haben, ist Ihnen sicher aufgefallen, daß mehr als 255 Weichen gesteuert werden können, nämlich 256; eine fehlt noch in der Rechnung! Wie bei den Lokomotiven benutzt der Rechner auch die 0 als Befehl; hier ist die gesuchte Weiche versteckt. Geben Sie in der Anweisung

```
PRINT#2,CHR$(33);CHR$(0);
```

ein, so schaltet die Weiche 256 nach links.





Interfacesteuerung mit Komfort

Damit die Bedienung des Märklin Interfaces 6050 komfortabler wird, wird hier eine in Maschinensprache geschriebene Routine vorgestellt, die vom Märklin Software-Service auf Anforderung geliefert wird. Die Routine unterstützt nur die Bedienung des Interfaces

6050, kann also nicht als allg. RS232-Treiber eingesetzt werden.

Die im C64 vorhandene RS232-Unterstützung wird durch dieses Programm ersetzt. Damit wird der vom C64 benötigte RS232-Puffer von 512 Byte nicht angelegt.

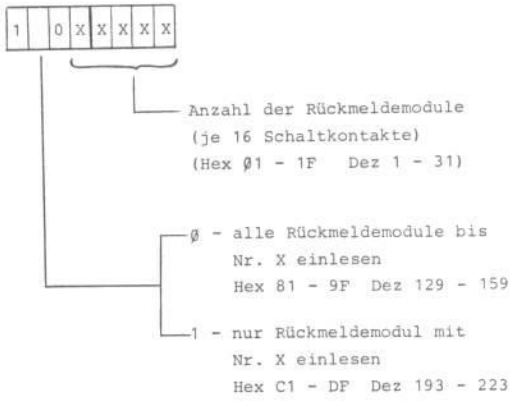
Alle zu übertragenden Bytes werden kontrolliert und eine eventuelle Fehlermeldung in ST ausgegeben. Die CTS-Behandlung erledigt die Routine selbständig. (befehlsweise!).

Die RM-Kontakte können jetzt bitweise mit der USR-Funktion und der Kontaktnummer abgefragt werden. Es wird nur das gerade abgefragte Bit gelöscht. Nicht abgefragte Kontakte werden beim nächsten Einlesen nicht überschrieben.

Das Programm benötigt den Speicherplatz von 49152 bis 49763 und die Adressen 2, 3, 248, 249 und den USR-Vektor.

```
Laden (von Hand)      LOAD "SYSOUT", 8,1
                      NEW
                      eigenes Programm laden oder eingeben

Laden im Programm     1 ON X GOTO 3
                      2 X = 1 : LOAD "SYSOUT", 8,1
                      3 OUT = 49152
                      4 REM hier gehts los

'CTS'      Die CTS-Leitung wird abgefragt und erst wenn das IF
           bereit ist zu empfangen, wird auch gesendet.

SYS 49152 , 1. byte
SYS 49152 , 1. byte , 2. byte

Allg. Aufruf der Maschinenroutine im C64 zur IF-Bedienung. Jeder mögliche IF-Befehl wird
gesendet. Ein- oder Zwei-byte werden erkannt und unterschiedlich behandelt. Die jeweils
letzte Nummer für Lok, Weiche oder SF-Modul wird getrennt gespeichert und wird bei
fehlendem 2.byte eingesetzt.
Jeder Einbyte-Befehl wird an das IF gesendet. ein 2.byte wird dann ignoriert.

SYS 49152      Letzter gesendeter Befehl wird erneut an das IF
               gesendet.
SYS 49152 , 96
SYS 49152 , 96 2. byte
               Der Befehl 'Freigabe'wird ohne CTS-Behandlung
               gesendet. Ein 2. byte wird ignoriert. Der
               Befehlspuffer zur Wiederholung wird nicht
               berührt, d.h. letzter Befehl vor Freigabe wird
               wiederholt

USR (kontakt)  Der USR-Vektor wird in der SYS-Routine gesetzt.
               Bei Übergabe der korrekten RM-Kontaktnummer 1 bis
               496 wird der eingelesene Wert übergeben.
               ausgelöst       = -1 oder wahr   /true
               nicht ausgelöst =  0 oder falsch /false

ST             Nach SYS-Aufruf kann in der C64-Statusvariablen ST
               eine Meldung über die korrekte Ausführung des
               Befehls gelesen werden.

               0 = korrekt ausgeführt
               1 = zu viele s88 angegeben
               2 = kein gültiger Befehl
               3 = RS232 Lesefehler
               4 = Nothalt (kein CTS frei innerhalb 5 sec.)
               5 und mehr = totales Chaos und Anarchie
                            (vom System)
```

Rückmeldebausteine einlesen



Beispiel 1

```
1 ON X GOTO 3
2 X = 1 : LOAD "SYSOUT", 8,1
3 OUT = 49152
10 REM warten auf kontakt 3
20 SYSOUT , 129
30 IF USR ( 3 ) THEN SYS 49152 : GOTO 30
40 REM reaktion auf kontakt 3
```

Beispiel 2

```
1 ON X GOTO 3
2 X = 1 : LOAD "SYSOUT", 8,1
3 OUT = 49152
10 REM nothaltbedienung ohne datenverlust
20 SYSOUT , BE , AD
30 IF ST = 4 THEN GOSUB 100
40 REM weiter geht's im programm
90 END
100 SYSOUT  , 96
110 FOR T = 0 TO 1000 : NEXT
120 SYSOUT  : IF ST = 4 THEN 100
130 RETURN
```

Betriebsanleitung zu Digi 7

Programmstart mit
Load "Digi 7", 8, 1 Return
RUN Return

Mit diesem Programm kann ein einfacher Fahrplan für eine kleine Modellanlage geschrieben werden. Die Befehle müssen fortlaufend numeriert entsprechend der unten angegebenen Syntax eingegeben werden. Zuerst wird die Befehlsnummer eingegeben, z.B. 1 Return.
Jetzt kommt die Eingabe des Befehls z.B. L 0 3 G 0 4 Return.
(Lok 3 Geschwindigkeit 4).
So kann ein bis zu 50 Befehle umfassender Fahrplan eingegeben werden.
Auf dem Bildschirm erscheint der jeweils bearbeitete Abschnitt des Fahrplans.

```
0 0 4 G 0 2     Lokbefehl       (Loknummer, Fahrstufe)
W 0 3 R         Weichenbefehl   (Weichennummer, Richtung)
R 0 1 K 1 2     Rückmeldebefehl (Modulnummer, Kontaktnummer)
```

Der Rückmeldebefehl wartet bis dieser Kontakt betätigt wird (Abbruch mit Taste S).

```
Z 4             4 sec Wartezeit
G 7             Sprung zu Fahrplanzeile 7
```

Um einen bereits geschriebenen Fahrplan speichern und edieren zu können, wurden weitere Befehle integriert. Die folgenden Befehle werden statt der Befehlsnummer eingegeben.

```
L    Return   Load Fahrplan
S    Return   Save Fahrplan
D    Return   Display Directory
K 4  Return   Kill 4 (Löschen von Zeile 4)
I 3  Return   Insert 3 (Einfügen zwischen Zeile 2 und 3)
```

Bei Insert muß als 2. Eingabe der einzufügende Befehl folgen. Bei Load und Save wird nach einem Filenamen gefragt.

Durch Eingeben einer bereits vorhandenen Zeilennummer können auch Zeilen überschrieben werden.

Mit G Return wird der Fahrplan gestartet und mit der Taste S kann er jederzeit abgebrochen werden.

```
  10  REM "

  20  REM "  * * * d i g i - s o f t * * *

  30  REM "  name: digi7      nr: 860901 kf

  40  REM "  copyright  c by 1986 maerklin

  50  REM "  letzte aenderung    870114 kf

  60  REM "

 100  OPEN "COM1:2400,N,8,2,CS10000,DS" AS #1
 110  DIM
         F$(500),
         B$(20):
      Z1$=SPACE$(3):
      Z2$=Z1$:
      Z0$=SPACE$(4)
 115  FL=0:
      FZ=1:
      FY=7:
      Y=2
 120  S=1:
      S1=1500:
      S2=500:
      S3=1000
 130  ROT=34:
      GRUEN=33:
      AUS=32:
      TM=.05
 140  DX=3:
      DY=50:
      D=1:
      MODE=1
 150  SX=60:
      SY=20
 300  CLS
 500  GOTO 4010

 700  REM "  zerlegen einer eingabe

 705  Z$=A$+"__"
 706  ZN=VAL(A$)
 710  WHILE ASC(Z$)<60 :
      |  Z$=MID$(Z$,2):
      WEND
```

```
715  A1$=LEFT$(Z$,1) :
     Z$=MID$(Z$,2)
720  A1=VAL(Z$)
725  WHILE ASC(Z$)<60 :
     |  Z$=MID$(Z$,2):
     WEND
730  A2$=LEFT$(Z$,1) :
     Z$=MID$(Z$,2)
735  A2=VAL(Z$)
741  IF
        A1>0
           THEN
              RSET Z1$=STR$(A1)
           ELSE
              RSET Z1$="    "
742  IF
        A2>0
           THEN
              RSET Z2$=STR$(A2)
           ELSE
              RSET Z2$="    "
745  A$=A1$+Z1$+" "+A2$+Z2$
760  RETURN

778  REM "  zerlegen im fahrplan

780  A1$=MID$(F$(PC),1,1):
     A1=VAL(MID$(F$(PC),2,3))
782  A2$=MID$(F$(PC),6,1):
     A2=VAL(MID$(F$(PC),7,3))
783  B$=A1$+A2$
784  RETURN

899  REM "  lesen der befehle in strings

900  RESTORE 6010:
     READ BZ:
     FOR I=1 TO BZ:
     |  READ B$(I):
     NEXT:
     RETURN

909  REM "  ausgabe der bildmaske fuer hauptmenue

910  RESTORE 6030:
     READ N:
     FOR I=1 TO N:
     |  READ PX,PY,A$:
     |  LOCATE PY,PX:
     |  PRINT A$;:
     NEXT:
     RETURN

919  REM "  ausgabe der bildmaske fuer load/save

920  RESTORE 6220:
     READ N:
     FOR I=1 TO N:
     |  READ PX,PY,A$:
     |  LOCATE PY,PX:
     |  PRINT A$;:
     NEXT:
     RETURN
```

```
 929  REM "  print fahrplan

 930  FOR I=1 TO 23 :
      |  J=FZ+I-12 :
      |  LOCATE I,Y
 931  |  IF
      |     J>0
      |        THEN
      |           RSET Z0$=STR$(J):
      |           PRINT Z0$;" : "; :
      |        ELSE
      |           PRINT SPACE$(5);
 932  |  IF
      |     J>0 AND J<=FL
      |        THEN
      |           PRINT F$(J);"     ":
      |        ELSE
      |           PRINT SPACE$(15);
 933  NEXT:
      RETURN

1000  REM "  lok-befehl

1005  PRINT #1,CHR$(A2);CHR$(A1);
1010  RETURN

1050  REM "  mag-befehl

1055  IF
         A2$="g"
            THEN
               MS=GRUEN
            ELSE
               MS=ROT
1060  PRINT #1,CHR$(MS);CHR$(A1);
1065  A1=TM:
      GOSUB 1205
1070  PRINT #1,CHR$(AUS);
1075  RETURN

1100  REM "  rm-befehl

1105  RM$=CHR$(A1+192)
1110  KN=16-A2:
      IF
         KN>7
            THEN
               K=2^(KN-8)
            ELSE
               K=2^KN
1115  PRINT #1,RM$;
1120  RB$=INPUT$(2,#1)
1125  IF
         KN<8
            THEN
               RB$=RIGHT$(RB$,1)
1130  KO=SGN(ASC(RB$) AND K)
1135  IF
         INKEY$="s"
            THEN
               S=1:
               RETURN
1140  IF
         KO=0
            THEN
               ( GOTO ) 1115
1145  RETURN

1150  REM "  neuer rm-befehl

1155  A1=A1-1:
      A2=A1AND15:
      A1=A1\16:
      RM$=CHR$(A1+192)
1160  KN=16-A2:
      IF
         KN>7
            THEN
               K=2^(KN-8)
            ELSE
               K=2^KN
1165  PRINT #1,RM$;
1170  RB$=INPUT$(2,#1)
1175  IF
         KN<8
            THEN
               RB$=RIGHT$(RB$,1)
1180  KO=SGN(AND ON(RB$) AND K)
1185  IF
         TAB(="s" TO S=1:
         RETURN
1190  IF
         KO=0
            THEN
               ( GOTO ) 1165
1195  RETURN

1200  REM "  zeit-befehl

1205  'SOUND S3,A1*18.2 :SOUND S2,1
1210  FOR I=1 TO A1*1250 :
      NEXT
1215  RETURN

1250  REM "  goto-befehl

1255  PC=A1 :
      MODE=3:
      IF
         S=1
            THEN
               CLS :
               S=0
1260  RETURN

1300  REM "  dir-befehl

1305  CLS
1310  FILES "*.fpl"
1315  INPUT A$
1320  CLS:
      GOSUB 910 :
      GOSUB 930
1325  RETURN
```

```
1350  REM "  save-befehl

1355  GOSUB 920
1360  INPUT "  filename ";N$
1365  N$=LEFT$(N$,8)+".fpl"
1370  OPEN"O",#2,N$
1375  WRITE #2,FL
1380  FOR I=1 TO FL:
      |  WRITE #2,F$(I) :
      NEXT
1385  CLOSE 2
1390  GOSUB 910
1395  RETURN

1400  REM "  load-befehl

1405  GOSUB 920
1410  INPUT "  filename ";N$
1415  N$=LEFT$(N$,8)+".fpl"
1420  OPEN "I",#2,N$
1425  INPUT#2,FL
1430  FOR I=1 TO FL:
      |  INPUT#2,F$(I) :
      NEXT
1435  CLOSE 2
1440  GOSUB 910
1450  REM "  insert-befehl

1455  FOR I=FL TO FZ STEP -1
1460  |  SWAP F$(I),F$(I+A1)
1465  NEXT
1470  FL=FL+A1
1475  RETURN

1500  REM "  delete befehl

1505  FOR I= FZ TO FL
1510  |  SWAP F$(I),F$(I+A1)
1515  NEXT
1520  FL=FL-A1 :
      GOSUB 930:
      RETURN

2000  REM "  lok-befehl

2010  IF
         A1<1 OR A1>80
            THEN
               RETURN
2020  IF
         A2<0 OR A2>31
            THEN
               RETURN
2030  SYNTAX=1 :
      RETURN

2050  REM "  mag-befehl

2060  IF
         A1<0 OR A1>255
            THEN
               RETURN
2070  SYNTAX=1 :
      RETURN

2100  REM "  rm-befehl

2110  IF
         A1<1 OR A1>31
            THEN
               RETURN
2120  IF
         A2<1 OR A2>16
            THEN
               RETURN
2130  SYNTAX=1 :
      RETURN

2150  REM "  zeit-befehl

2160  IF
         A1<0 OR A1>99
            THEN
               RETURN
2170  SYNTAX=1 :
      RETURN

2200  REM "  goto-befehl

2210  IF
         A1<0 OR A1>FL
            THEN
               RETURN
2220  IF
         A1=0
            THEN
               A1=1
2230  SYNTAX=1 :
      RETURN

2250  REM "  dir-befehl


2260  REM

2270  SYNTAX=1 :
      RETURN

2300  REM "  save-befehl


2310  REM

2320  SYNTAX=1 :
      RETURN

2350  REM "  load-befehl


2360  REM

2370  SYNTAX=1 :
      RETURN

2400  REM "  insert-befehl
```

```
2410  IF
         FZ<1 OR FZ>FL
            THEN
               RETURN
2420  IF
         A1<1 OR A1+FL>500
            THEN
               RETURN
2430  SYNTAX=1 :
      RETURN

2450  REM "  delete befehl

2460  IF
         FZ<1 OR FZ+A1>500
            THEN
               RETURN
2470  IF
         A1<1 OR A1>FL
            THEN
               RETURN
2480  SYNTAX=1 :
      RETURN

3000  REM

3010  PRINT #1,CHR$(193);:
      A$=INPUT$(2,#1) :
      PRINT ASC(A$);ASC(RIGHT$(A$,1))
3020  GOTO 3000

3030  REM "  test syntax

3040  SYNTAX=0:
      B$=A1$+A2$
3050  FOR I=1 TO BZ
3060  |  IF
      |     B$(I)<>B$
      |        THEN
      |           ( GOTO ) 3090
3070  |  ON
      |     I
      |        GOSUB 2010,2060,2060,2110,2160,2210,2260,2310,2360,2410,2460
3080  |  I=BZ
3090  NEXT I:
      RETURN
3100  ON
         I
            GOSUB 1005,1055,1055,1105,1205,1255,1305,1355,1405,1455,1505

4000  REM "  hauptprogrammschleife

4010  GOSUB 900 :

      REM "  befehle einlesen

4020  GOSUB 910 :

      REM "  hauptbild zeichnen

4030  GOSUB 930 :

      REM "  fahrplan zeichnen
```

```
4040  IF
         MODE=2
            THEN
               LOCATE 12,FY
            ELSE
               LOCATE DX,DY
4041  PRINT SPACE$(15)
4042  IF
         MODE=2
            THEN
               LOCATE 12,FY
            ELSE
               LOCATE DX,DY
4050  INPUT A$
4060  GOSUB 705 :

      REM "  eingabe zerlegen

4065  IF
         A1$="x"
            THEN
               MODE=1:
            ELSE
               IF
                  A1$="e"
                     THEN
                        MODE=2
4066  IF
         ZN>0
            THEN
               FZ=ZN
4067  IF
         A1$="h"
            THEN
               GOSUB 7000
4070  'IF ZN<0 THEN D=1:PC=0 ELSE IF ZN>0 THEN D=0:FZ=ZN
4080  GOSUB 3030 :

      REM "  syntax überprüfen

4090  IF
         SYNTAX=0
            THEN
               ( GOTO ) 4030
4100  IF
         MODE=1
            THEN
               GOSUB 5000 :
               GOTO 4020
4110  F$(FZ)=A$:
      IF
         FZ>FL
            THEN
               FL=FZ
4120  FZ=FZ+1:
      GOTO 4030

5000  REM "  fahrplanabarbeitung

5010  B$=A1$+A2$
5020  'stop
5030  II=0:
```

```
      FOR I=1 TO BZ
5040  |  IF
      |     B$(I)=B$
      |        THEN
      |           II=I :
      |           I=BZ
5050  NEXT I:
      IF
         II=0
            THEN
               MODE=1
5060  IF
         MODE=3
            THEN
               PC=PC+1:
               IF
                  PC>FL
                     THEN
                        MODE=1
5070  'stop
5080  ON
         II
            GOSUB 1005,1055,1055,1105,1205,1255,1305,1355,1405,1455,1505
5100  'stop
5110  IF
         INKEY$="s" OR S=1
            THEN
               MODE=1 :
               CLS
5115  IF
         MODE=1
            THEN
               RETURN
5120  GOSUB 780
5130  PRINT PC;F$(PC)
5140  GOTO 5020

6000  REM "  datenzeilen fuer befehle

6010  DATA 11,lg,mg,mr,rk,z_,g_,di,s_,l_,i_,d_,

6020  REM "  datenzeilen fuer hauptmenue

6030  DATA 21
6040  DATA 40,4,----------------------------
6050  DATA 40,5,-- edit-mode-  e           --
6060  DATA 40,6,----------------------------
6070  DATA 40,7,-- lok      -  l 15 g 4    --
6080  DATA 40,8,-- weiche   -  m 4 r       --
6090  DATA 40,9,-- kontakt  -  r 30 k 14   --
6100  DATA 40,10,-- zeit     -  z 13        --
6110  DATA 40,11,-- goto     -  g 15        --
6120  DATA 40,12,----------------------------
6130  DATA 40,13,-- exit-edit   x          --
6140  DATA 40,14,----------------------------
6150  DATA 40,15,-- insert   -  i 14        --
6160  DATA 40,16,-- delete   -  d 3         --
6170  DATA 40,17,-- load     -  l           --
6180  DATA 40,18,-- save     -  s           --
6190  DATA 40,19,-- dir      -  di          --
6200  DATA 40,20,----------------------------
6201  DATA 40,21,-- abbruch     s           --
6202  DATA 40,22,----------------------------
```

```
6203  DATA 40,23,-- H I L F E   h          --
6204  DATA 40,24,----------------------------

6210  REM "  datenzeilen fuer save/load

6220  DATA 1
6230  DATA 40,1 ,""
6240  DATA 1
6250  DATA 40,1 ,          ""
6900  LOCATE 23,15,1
6910  PRINT" WEITER MIT IRGENDEINER TASTE ";:Q$=INPUT$(1)
6940  CLS
6945  PRINT""
6950  RETURN
7000  CLS:PRINT
7010  PRINT"   Das Programm DIGI 7 besitzt 2 Betriebsarten. Nach dem Starten des"
7020  PRINT"   Programms befinden Sie Sich im DIREKT-MODE."
7030  PRINT"   Im DIREKT-MODE werden alle Befehle  die eingegeben werden sofort aus-"
7040  PRINT"   geführt. Damit können Sie ihre Verbindung zum INTERFACE 6050 austesten"
7050  PRINT"   und verschiedene Versuche durchführen."
7060  PRINT""
7070  PRINT"   z.B.  :      m 2 g        ( Magnetartikel 2 gerade/grün        )"
7080  PRINT"               r 1 k 4      ( Warte bis Kontakt 4 am Modul 1"
7090  PRINT"                              betätigt wird                  )"
7100  PRINT"               l 10 g 16    ( Einschalten der Sonderfunktion"
7110  PRINT"                              an der  Lok 10                 )"
7120  PRINT"               l 10 g 20    ( Lok 10  Fahrstufe  4  Sonderf. an  )"
7130  PRINT"               l 10 g 4     ( Lok 10  Fahrstufe  4  Sonderf. aus )"
7140  PRINT"               l 10 g 15    ( Lok 10  Wenden        Sonderf. aus )"
7150  PRINT"               l 10 g 9     ( Lok 10  Fahrstufe  9  Sonderf. aus )"
7160  PRINT"               l 10 g 30    ( Lok 10  Fahrstufe 14  Sonderf. an  )"
7170  PRINT"               l 10 g 16    ( Lok 10  Stop          Sonderf. an  )"
7180  PRINT"               z 5          ( Zeitverzögerung um 5 Sekunden      )"
7190  GOSUB 6900
7200  PRINT"   Durch weitere Befehle im DIREKT-MODE kann mann Fahrpläne laden,
        speichern"
7210  PRINT"   und ausführen."
7220  PRINT""
7230  PRINT"   z.B.        l            ( Laden eines Fahrplanes             )"
7240  PRINT"               s            ( Speichern eines Fahrplanes         )"
7250  PRINT"               di           ( Anzeigen aller auf Disk vorhandenen"
7260  PRINT"                              Fahrpläne                          )"
7270  PRINT"               g 1          ( Starten des momentan vorhandenen"
7280  PRINT"                              Fahrplanes ab der Zeile 1          )"
7290  PRINT""
7300  PRINT"   Die Befehle l und s fordern Sie noch zusätzlich auf, den Namen eines"
7310  PRINT"   Fahrplanes einzugeben. Beim Befehl g muss die 'zeile' sofort angegeben"
7320  PRINT"   werden, ab der der Fahrplan ablaufen soll."
7330  PRINT"   Aber um einen Fahrplan ablaufen zu lassen, muss er zuerst erstellt
        werden."
7340  PRINT"   Dazu existiert der EDIT-MODE. Ein Überwechseln vom DIREKT-MODE in den"
7350  PRINT"   EDIT-MODE ist durch Eingabe des Befehls e möglich."
7360  PRINT""
7370  PRINT"   z.B.        e            ( Wechsel in den EDIT-MODE           )"
7380  GOSUB 6900
7400  PRINT"   Im EDIT-MODE können Sie die Befehle l, m, z, r, und g verwenden. Sobald"
7410  PRINT"   die Eingabe einer Zeile abgeschlossen ist, wird sie überprüft und in
        den"
7420  PRINT"   Fahrplan eingetragen. Der Rechner erwartet dann die Eingabe der
        nächsten"
7430  PRINT"   Zeile. Wenn ihr Fahrplan fertig ist, verlassen Sie den EDIT-MODE mit"
7440  PRINT"   dem Befehl x ."
7450  PRINT"   Der EDIT-MODE bietet aber noch weitere Möglichkeiten. Geben Sie zum"
```

```
 7460  PRINT"   Beispiel keinen Befehl sondern eine Zahl ein, so wird der Editor an die"
 7470  PRINT"   Zeile im Fahrplan springen und dort die nächste Eingabe erwarten."
 7480  PRINT""
 7490  PRINT"   Ein Teil der Editor-Befehle sind auch aus dem DIREKT-MODE verwendbar."
 7500  PRINT"   Wird im DIREKT-MODE nur eine Zahl eingegeben, so wirkt dies wie ein
           LIST"
 7510  PRINT"   in BASIC. NUR aus dem DIREKT-MODE anwendbar sind die Befehle i und d,
          die"
 7520  PRINT"   eine Zeile in den Fahrplan einfügen bzw. löschen."
 7530  PRINT"   Zuerst muss die Zeilennummer eingegeben werden, bei der die Änderung"
 7540  PRINT"   efolgen soll. Dann stellt der Rechner den angegebenen Abschnitt dar."
 7550  PRINT"   Jetzt folgt der eigentliche Befehl i oder d mit der angabe ,wie viele"
 7555  PRINT"   Zeilen eingefügt oder entfernt werden sollen."
 7560  PRINT""
 7570  PRINT"   z.B.       20              ( Bei Zeile 20 editieren        )"
 7580  PRINT"              i 2             ( 2 Zeilen einfügen             )"
 7590  GOSUB 6900
 7600  PRINT"   VORSICHT   bevor Sie Ihren ersten grösseren Fahrplan schreiben,"
 7610  PRINT"              sollten Sie Sich gründlich mit dem Editor auskennen."
 7620  PRINT"   DENN       es ist sehr ärgerlich, wenn der schöne 250-Zeilen-"
 7630  PRINT"              Fahrplan durch eine Fehleingabe zerstört wird."
 7640  PRINT"   DESSHALB   sollten Sie auch immer wieder von der Möglichkeit"
 7650  PRINT"              gebrauch machen, den Fahrplan zu speichern."
 7660  PRINT""
 7670  PRINT"   TROTZDEM   ist noch nicht alles verloren, wenn das Programm mit"
 7680  PRINT"              einer Fehlermeldung abbricht. Kein Problem. Bewegen Sie"
 7690  PRINT"              den Cursor in die linke obere Ecke und geben Sie die"
 7700  PRINT"              folgenden Befehle ein."
 7710  PRINT""
 7720  PRINT"              cls:goto  300:"
 7730  PRINT""
 7740  PRINT"              Allerdings wäre es ratsam dann so schnell wie möglich"
 7750  PRINT"              den Fahrplan zu speichern."
 7790  GOSUB 6900
 7990  GOSUB 910
 8000  RETURN
```

Befehlsbeschreibung für den ATARI 520ST

Der serielle Anschluß befindet sich beim Atari 520ST auf der Rückseite des Geräts zwischen dem Druckeranschluß und dem Anschluß für das Diskettenlaufwerk. Der 25-polige RS232 Stecker ist mit MODEM bezeichnet. Um ein Kabel löten zu können, müssen Sie eine 25-polige D-Sub-Kupplung mit Gehäuse und Zugentlastung besorgen. Bitte achten Sie vor dem Löten darauf, daß diese Kupplung auch problemlos auf den Stecker Ihres Rechners paßt und nicht irgendwelche Teile der Kupplung stören.

1. Anschluß von Märklin Interface und ATARI 520ST

Recive Data	(1)	auf	(2)	Transmit Data	TXD
Transmit Data	(4)	auf	(3)	Recive Data	RXD
Clear to Send	(5)	auf	(5)	Clear to Send	CTS
Ground	(3)	auf	(7)	Signal Ground	GND
Ground	(3)	auf	(1)	Protective Ground	GND
			(8)	DCD Brücke auf (20)	DTR

Anschlußbild!

2. Einstellung am Interface:

ON			
1	2	3	4
		OFF	OFF

3. Initialisierung der Schnittstelle:
Definieren der Startadresse für PEEK und POKE

```
Startadresse
für PEEK & POKE                                    DEF SEG = &HFFFA00

Übertragungsparameter einstellen:        POKE 37,8: POKE 41,152
Baudrate  (2400)
Datenbits (8)
Parität   (keine)
Stoppbits (2)

4. Befehle zur direkten und Programmsteuerung
CTS-Statusleitung abfragen:

Schleife solange CTS belegt                    WHILE PEEK(1) AND 4: WEND
Parallelport einlesen
Verknüpfung (logisch UND)
Verknüpfungsmaske (Bit 3)

Dieses Statussignal muß vor der Übertragung jedes Befehls getestet werden.  Es zeigt an,
ob das Interface empfangsbereit ist.
Lokokmotive steuern:
Ausgabe auf Kanal 1
Fahrstufe   (0-31)                             OUT 1,DB
Lok-Adresse (1-80)                             OUT 1,AB

Magnetartikel stellen:
Magnetartikelstellung (33,34)                  OUT 1,DB
Magnetartikeladresse (1-255,0)                 OUT 1,AB

Magnetartikelantriebe abschalten:              OUT 1,32

Magnetartikelstrom des zuletzt gestellten Magnetartikels abschalten

NOTHALT (Taste stop):
Spannungsversorgung der Modellbahn-Anlage sofort abschalten      OUT 1,97

FREIGABE (Taste go):
Spannungsversorgung der Modellbahn-Anlage wiedereinschalten      OUT 1,96

Dieser Befehl kann zu jeder Zeit an das Interface gesendet werden,  selbst wenn die CTS-
Leitung  keine Empfangsbereitschaft des Interface anzeigt z.B.  nach einem NOTHALT  oder
Kurzschluß. Nach dem Wiedereinschalten der Spannungsversorgung wird dann der während des
Kurzschlusses eingegangene Befehl abgearbeitet.

Befehle zur Steuerung der Rückmeldemodule:       OUT 1,128+M
Einlesen mehrerer Rückmeldemodule:
Einlesen ab Modul Nr. 1
         bis Modul Nr. M

Einlesen eines bestimmten Rückmeldemoduls:       OUT 1,192+M
Einlesen eines Moduls
         Modul Nr. M

Reset-Mode einschalten:
RESET Ein                    OUT 1,192
(Standardeinstellung des Interface nach dem Einschalten)
Reset-Mode ausschalten:
RESET Aus                    OUT 1,128

Bei  jedem Einlesen werden die Rückmeldemodule vom Interface automatisch auf 0
zurückgesetzt.  Dieser Reset wird durch den Befehl 128 ausgeschaltet.  Mit 192 kann  das
Interface  wieder in den Reset-Mode gebracht werden.  Keiner dieser beiden Befehle führt
direkt zu einem Reset der Module.
```

Befehle, um angeforderte Rückmeldungen zu empfangen und auszuwerten

```
Empfang von 1 Byte vom Interface:        WHILE INP(-1)=0:WEND:B(I)=INP(1)
Warteschleife
Input-Status lesen
Kanal 1 (RS 232)
Test ob Empfangspuffer leer
Feldvariable für Rückmeldebytes
Nummer des Bytes
Byte abholen
Kanal 1

Schnelles Auswerten eines einzelnen Kontaktes     K=SGN(B(KN\8) AND M(KN MOD 8)
Kontaktstellung
Ergebnis auf 0 oder 1 setzen
Feldvariable mit Rückmeldebytes
Berechnung der Bytenummer (0 bis N-1)
Verknüpfung (logisch UND)
Feldvariable mit Verknüpfungsmaske
Auswahl der Verknüpfungsmaske (0-7)
```

KN ist die Kontaktnummer minus 1 (für 2 Rückmelde-Module: 0-31)

Als Vorbereitung für diese schnelle Auswertung nuß bei der Initialisierung des Programms folgende Zahlen stehen, die ein Feld dimensioniert und in jeder Variablen ein anderes Bit setzt. Durch eine Verknküpfung (logisch UND) mit diesen Variablen kann sehr schnell ein einzelnes Bit aus einem Byte isoliert und ausgewertet werden.

```
DIM M(7):FOR I=0 TO 7:M(I)=2^(7-I):NEXT
```

Auswerten aller Kontakte und speichern in einem zweidimensionalen Feld

```
FOR I=0 TO N-1
          B% = B%(I)
          FOR J=7 TO 0 STEP -1
                    S% (I,J) = B% MOD 2
                    B% (I)   = B%  \  2
          NEXT J
NEXT I
```

Die Variable N gibt die Anzahl der gelesenen Bytes an. In der ersten Schleife (I) wird jede Zahl in B geschrieben aus der dann jeweils Kontakte berechnet werden können. In der zweiten Schleife (J) wird die Zahl B zerlegt, die Kontaktstellung ermittelt und im Feld S gespeichert.

```
DIM B%(61), S%(61,7)
```

Reserviert den notwendigen Speicherplatz für die Felder B und S. Der DIM-Befehl muß am Anfang des Programms stehen.

Anschlußbeschreibung für den IBM PC und den Amiga

Der serielle Anschluß findet sich beim IBM PC und kompatiblen Rechnern auf einer Schnittstellenkarte, entweder einer seriellen Schnittstellenkarte oder einer Multi I/O-Karte. Um ein Kabel löten zu können, müssen Sie eine 25-polige D-Sub-Kupplung mit Gehäuse und Zugentlastung besorgen. Bitte achten Sie vor dem Löten darauf, daß diese Kupplung auch problemlos auf den Stecker Ihres Rechners paßt und nicht irgendwelche Teile der Kupplung stören.
Die Steckerbelegung und die BASIC-Befehle sind für den Amiga dieselben wie für den IBM PC.

```
1. Anschluß von Märklin Interface   und     IBM PC
                Receive Data    (1) auf (2) Transmit Data      TXD
                Transmit Data   (4) auf (3) Receive Data       RXD
                Clear to Send   (5) auf (5) Clear to Send      CTS
                Ground          (3) auf (7) Signal Ground      GND
```

```
                Ground          (3) auf (1) Protective Ground  GND
                                            (6,8) Brücke auf (20)DTR
Anschlußbild!

2. Einstellung am Interface:       ON ON
                                    1  2  3   4
                                         off off

3. Initialisierung der Schnittstelle:
                   OPEN "COM1: 2400, N, 8, 2, CS10000, DS" AS #1
Definieren der Startadresse für PEEK und POKE

Kanal öffnen
serielle Schnittstelle COM1
Baudrate (2400)
Parität (keine)
Stoppbits (2)
CTS bedienen (Time out nach 10 Sekunden)
CTS nicht bedienen
Eröffnen als I/O Kanal 1

4. Befehle zur Steuerung

Lokomotive steuern:
                   PRINT #1, CHR$(DB)+CHR&(AB);
Ausgabe auf Kanal 1
Fahrstufe   (0-31)
Lok-Adresse (1-80)

Magnetartikel stellen:
                   PRINT #1, CHR$(DB)+CHR$(AB);
Magnetartikelstellung (33,34)
Magnetartikeladresse (1-255,0)

Magnetartikelantriebe abschalten:
                   PRINT #1, CHR$(32);

Magnetartikelstrom des zuletzt gestellten Magnetartikels abschalten

NOTHALT (Taste stop):
                   PRINT #1, CHR$(97);
Spannungsversorgung der Modellbahn-Anlage sofort abschalten

FREIGABE (Taste go):
                   PRINT #1, CHR$(96);
Spannungsversorgung der Modellbahn-Anlage wiedereinschalten

Dieser Befehl kann zu jeder Zeit an das Interface gesendet werden,  selbst wenn die CTS-
Leitung  keine Empfangsbereitschaft des Interface anzeigt z.B.  nach einem NOTHALT  oder
Kurzschluß. Nach dem Wiedereinschalten der Spannungsversorgung wird dann der während des
Kurzschlusses eingegangene Befehl abgearbeitet.

Befehle zur Steuerung der Rückmeldemodule:
Einlesen mehrerer Rückmeldemodule:
                    PRINT #1, CHR$(128+M);
Einlesen ab Modul Nr. 1
         bis Modul Nr. M

Einlesen eines bestimmten Rückmeldemoduls:
                    PRINT #1, CHR$(192+M);
Einlesen eines Moduls
         Modul Nr. M
```

```
Reset-Mode einschalten:
                    PRINT #1, CHR$(192);
RESET Ein
(Standardeinstellung des Interface nach dem Einschalten)
Reset-Mode ausschalten:
                    PRINT #1, CHR$(128);
RESET Aus
```

Bei jedem Einlesen werden die Rückmeldemodule vom Interface automatisch auf 0 zurückgesetzt. Dieser Reset wird durch den Befehl 128 ausgeschaltet. Mit 192 kann das Interface wieder in den Reset-Mode gebracht werden. Keiner dieser beiden Befehle führt direkt zu einem Reset der Module.

Befehle, um angeforderte Rückmeldungen zu empfangen und auszuwerten

```
Empfang von N Byte vom Interface:
                     B$=INPUT$(N,#1);
Stringvariable
erwarte Byte aus Datenkanal
Anzahl der Bytes N
(2 Bytes pro angefordertem Rückmeldemodul)
Nr. des Datenkanals
```

```
Schnelles Auswerten eines einzelnen Kontaktes
                         K=SGN(B(KN/8)AND M(KN MOD 8)
Kontaktstellung
Ergebnis auf 0 oder 1 setzen
Feldvariable mit Rückmeldebytes
Berechnung der Bytenummer (0 bis N-1)
Verknüpfung (logisch UND)
Feldvariable mit Verknüpfungsmaske
Auswahl der Verknüpfungsmaske (0-7)
```

KN ist die Kontaktnummer minus 1 (für 2 Rückmelde-Module: 0-31)

Als Vorbereitung für diese schnelle Auswertung nuß bei der Initialisierung des Programms folgende Zahlen stehen, die ein Feld dimensioniert und in jeder Variablen ein anderes Bit setzt. Durch eine Verknküpfung (logisch UND) mit diesen Variablen kann sehr schnell ein einzelnes Bit aus einem Byte isoliert und ausgewertet werden.

Auswerten aller Kontakte und speichern in einem zweidimensionalen Feld

```
        FOR I=0 TO N-1
                  B% = B%(I)
                  FOR J=7 TO 0 STEP -1
                            S% (I,J) = B% MOD 2
                            B% (I)  = B%  \  2
                  NEXT J
        NEXT I
```

Die Variable N gibt die Anzahl der gelesenen Bytes an. In der ersten Schleife (I) wird jede Zahl in B geschrieben aus der dann jeweils Kontakte berechnet werden können. In der zweiten Schleife (J) wird die Zahl B zerlegt, die Kontaktstellung ermittelt und im Feld S gespeichert.

```
         DIM B%(61), S%(61,7)
```

Reserviert den notwendigen Speicherplatz für die Felder B und S. Der DIM-Befehl muß am Anfang des Programms stehen.

Grundlegende Steuerungen

Anschluß der Geräte

Die Ergänzung einer Digital-HO-Anlage um einen Computer ist einfach. Das Interface ist links an die Central Unit oder das äußerst linke Fahrgerät (Control 80) anzuschließen und mit der seriellen Schnittstelle des Computers (beim C64 dem User Port) zu verbinden.

Das Interface kann zwar direkt an die Zentraleinheit angeschlossen werden, um die Modellbahnanlage ausschließlich mit dem Computer zu betreiben, für das "Teststadium" empfiehlt sich jedoch der Anschluß eines Fahrreglers Control 80 um einen sichernen Not-Ausschalter in das System integriert zu haben. Wer vorhat ausschließlich mit dem Computer seine Anlage zu steuern findet eine preiswerte Steuereinheit in der Märklin Digital HO Grundpackung. An diese Gerät das Zentraleinheit, und mit eingeschränkten Funktionen Fahrpult und Stellpult vereint, kann ebenfalls das Interface angeschlosssen werden.

Die Zentraleinheit (Central Unit) des digitalen Steuersystems kann sowohl Daten vom Computer als auch von den anderen angeschlossenen Geräten verarbeiten, so daß die Tasten "stop" und "go" weiterhin funktionieren.

Das Digitalsystem muß zuerst eingeschaltet werden und anschließend der Computer.

Zusammenfassung der Bytes

Das Interface arbeitet mit Information die aus einem oder zwei Bytes bestehen können. Das zweite Byte wird generell für eine Nummer (Loknummer, Weichendecodernummer und später auch Zusatzfunktionspultnummer) herangezogen.

Entscheidend ist das erste Byte, da mit ihm die Auswahl getroffen wird, was zu steuern ist. Einen Überblick gibt folgenden Tabelle:

Übersicht über das erste Byte der Steuerinformation

Dez.	Binär Bit: 76543210	Bereich	Wirkung (ZF=Zusatzfunktionen)	
0	00000000	Loksteuerung	Stand	ZF:aus
1	00000001	Loksteuerung	Fahrstufe 1	ZF:aus
:	:	:	:	:
14	00001110	Loksteuerung	Fahrstufe 14	ZF:aus
15	00001111	Loksteuerung	Umschalten	ZF:aus
16	00010000	Loksteuerung	Stand	ZF:ein
17	00010001	Loksteuerung	Fahrstufe 1	ZF: ein
:	:	:	:	:
30	00011110	Loksteuerung	Fahrstufe 14	ZF:ein
31	00011111	Loksteuerung	Umschalten	ZF:ein
32	00100000	Magnetartikel	ausschalten	
33	00100001	Magnetartikel	grün,gerade,etc.	
34	00100010	Magnetartikel	rot,abbiegen,etc.	
35	00100011	zur Zeit nicht	genutzt	
:	:	:		

Dez.	Binär Bit:76543210	Bereich	Wirkung (M. = Modul)
63	00111111	zur Zeit nicht	genutzt
64	01000000	Sonderfunktionen	(geplant)
:	:	:	
79	01001110	Sonderfunktionen	(geplant)
80	01010000	zur Zeit nicht	genutzt

:	:	:
95	01011111	zur Zeit nicht genutzt
96	01100000	Freigabe
97	01100001	Nothalt
98	01100010	zur Zeit nicht genutzt
:	:	:
127	01111111	zur Zeit nicht genutzt
128	10000000	Rückmeldung RESET-Mode ausschalten
129	10000001	Rückmeldung M. bis Nr. 1 einlesen
:	:	: : : : :
191	10111111	Rückmeldung M. bis Nr. 63 einlesen
192	11000000	Rückmeldung RESET-Mode einschalten
193	11000001	Rückmeldung nur M. Nr. 1 einlesen
:	:	: : : :
255	11111111	Rückmeldung nur M. Nr. 63 einlesen

Aus dieser Tabelle geht hervor, daß wiederum die erste Hälfte des ersten Bytes die Art der Steuerung übernimmt und die zweite Hälfte (Bits 0 bis 3) mehr oder weniger wieder eine Nummernfunktion aufweist wie das zweite Byte.

Die Reihenfolge der Bits von links nach rechts entsprechend, wollen wir die Tabelle vom unteren Ende her besprechen. Das erste Bit wählt also die Rückmeldebausteine aus, und wenn dieses gesetzt ist, ist das zweite Bit für die Information 'nur'oder 'bis'Modulnummer einlesen zuständig. Die restlichen sechs Bit haben lediglich Nummernfunktionen. Wichtig ist noch das alleinige Setzen des ersten Bits (Bit 7), da hier ein Rücksetzen der Rückmelde-Module erfolgt.

Das Setzen von Bit 6 (natürlich ohne Bit 7) hat im momentanen Ausbauzustand von Digital-Anlagen nur in zwei Fällen Bedeutung, nämlich beim Nothalt (Simulierung der 'stop'-Taste) und bei der Freigabe (Simulierung der 'go'-Taste). In Zukunft soll dieses Bit für Sonderfunktionspulte zuständig sein, wobei im Zusammenhang mit den Bits 0 bis 3 bis zu vier verschiedene Funktionen angewählt werden können. Außerdem kann im zweiten Byte eine Sonderfunktionsadresse zwischen 1 und 80 gewählt werden, so daß man davon ausgehen kann, daß bei den Sonderfunktionen die gleiche Decodiertechnik wie bei den Lokmodulen zu Grunde liegt.

Bit 5 ist für die Weichen- und Signalschaltung sowie zur Steuerung anderer Magnetartikel vorgesehen, wobei hier von den 31 Möglichkeiten jedoch nur drei genutzt werden. Die restlichen 28 Möglichkeiten sind zur Zeit ebenfalls ungenutzt.

Bei den Bits 5 bis 7 war es bisher so, daß alle nachfolgenden Bits ihre Bedeutung nur erlangten, wenn eines der Bits 5, 6 oder 7 gesetzt war. Sind keine dieser Bits gesetzt, so ist Bit 4 für die Zusatzfunktion der Lok zuständig, wobei in diesem Zusammenhang auch die Geschwindigkeit und der Richtungswechsel einer Lok inbegriffen sind. Die Geschwindigkeit ist dabei jedoch unabhängig vom Zustand des Bits 4, wobei das Setzen aller Bits von 0 bis 3 eine Umschaltung der Richtung bedeutet.

Damit ist der gesamte Rahmen der Steuerinformation über Bits beschrieben. Wer jedoch weiter in die Technik des Interfaces einsteigen möchte, für den haben wir zusätzliche Informationen.

Die Schnittstelle zwischen Computer und Interface besteht aus fünf Leitungen:

- Daten vom Computer zum Interface (TXD)
- Daten vom Interface zum Home-Computer (RXD)
- Steuerleitung zur Identifikation der Empfangsbereitschaft des Interfaces (DSR)
- + 5V (Spannungsversorgung)
- Masse (Ground)

Wer sich die Rückseite des Interfaces näher betrachtet, wird dabei vier DIL-Schalter finden, die es ermöglichen, die drei Datenleitungen einzeln von negative auf positive Logik umzuschalten.

Näheres ist aus folgender Tabelle ersichtlich:

Schalter	Leitung	in (on)	aus (off)
1	TXD	negativ	positiv
2	RXD	negativ	positiv
3	DSR	negativ	positiv

Dem vierten Schalter kommt im Moment keine Bedeutung zu.

Steuerung einer Lok

Zur Steuerung einer Lok beginnt man am besten auf einem einfachen Gleisovals. Für die ersten Versuche reicht ein kleiner Gleiskreis aus. Sofern Sie eine größere Anlage besitzen, sollten Sie die Weichen so stellen, daß eine Lok ungehindert auf einem geschlossenen Kreis fahren kann.

Fahrt

Im Gegensatz zu den normalen Transformatoren mit ihren Fahrreglern, die eine analoge Fahrspannung ausgeben (stufenlos) ist die Spannung an den Gleisen bei der Digital-Anlage konstant. Die Steuereinheit arbeitet mit diskreten Signalen. Es gibt 15 Fahrstufen, einen Befehl um die Fahrtrichtung umzusteuern und einen Befehl um ene Zusatzfunktion zu aktivieren.

```
10 REM "
20 REM "  * * * d i g i - s o f t * * *
30 REM "  name: digi1      nr: 860826 kf
40 REM "  copyright  c by 1986 maerklin
50 REM "  letzte aenderung    860826 kf
60 REM "
100 OPEN "COM1:2400,N,8,2,CS10000,DS" AS #1
110 CLS
200 INPUT " loknr.  (1-80)  ";LN
210 INPUT " geschw. (0-15)  ";GE
300 PRINT #1,CHR$(GE);CHR$(LN);
310 GOTO 200
```

Zunächst wird die Schnittstelle des Rechners aktiviert, um mit dem Interface Daten austauschen zu können. Die Ausgabe von Daten an das Interface kann man auch als sequentielle Datei betrachten.

Sollten Sie bei einer Ausgabe auf den Fehler 'FILE NOT FOUND ERROR'treffen, so haben Sie mit Sicherheit den Open-Befehl vergessen. Der OPEN-Befehl muß natürlich auch gegeben werden, wenn Sie Eingaben im Direktmodus durchführen.

Für Steueraufgaben wie beschleunigen, bremsen oder warten sollte die Uhr des Rechners herangezogen werden. Je nach Computersprache und Rechnertyp wird die Uhr mit einem anderen Befehl aufgerufen. Es empfiehlt sich vor Beginn einer Operation die Zeit zunächst einer anderen Variablen zu übergeben, und jeweils die Differenz einer aktuellen Zeit zu der zwischengespeicherten Zeit zu überprüfen. Diese Methode ist universeller als Zeitverzögerungen im Programmablauf durch leere FOR...NEXT-Schleifen zu erzeugen. Wer mit einem IBM PC kompatiblen rechner arbeitet braucht seine Programm so nur einmal zu entwickeln. Ob Standard PC oder 80386 System mit 20 MHz Taktfrequenz,ob mit Interpreter oder Compiler, die Programme bleiben immer gleich.

Als nächstes folgen die Geschwindigkeit und die Loknummer in dezimaler Schreibweise. Das Zeichen wird im sogenannten ASCII-Code (American Standard Code of Information Interchange) dargestellt.

Da der Bereich der ASCII-Zeichen von 0 bis 31 mit Steuerzeichen belegt ist, und diese nur umständlich im PRINT-Befehl anzugeben wären, wird der Befehl CHR$() herangezogen, der eine entsprechende Umwandlung vornimmt.

Die erste Ziffer gibt die Geschwindigkeit, und die zweite Ziffer die Loknummer an. Merke: Mit PRINT #2, CHR$(X) CHR$(Y) wird eine Lok mit der Nummer Y auf eine Geschwindigkeit X gesetzt.

Damit die Lok nicht endlos weiterfährt, wollen wir sie wieder stoppen, aber erst, nachdem sie 10 Sekunden gefahren ist.

Wie bereits erwähnt, wird die Zeit seit Einschalten des Computers in der Variablen TI in 1/60 Sekunden gezählt. In Turbo BASIC gibt es den Befehl DELAY, in Standard GW-BASIC den Befehl TIMER. Zehn Sekunden entsprechen also 600/60 Sekunden. Man kann die Zeit, die seit Besetzen der Variablen T verstrichen ist, durch die Differenz TI-T ermitteln.

```
REM Programm in QickBASIC
SUB LOK (LN,VS) STATIC
REM Lokomotive steuern

        PRINT #1,CHR$(VS);CHR$(LN);
END SUB
REM Steuerprogramm für eine Lokomotive mit QuickBASIC
REM Interface initialisieren
        OPEN "COM1: 2400, N,8,2,CS10000, DS" AS #1

CLS:REM BILDSCHIRM LÜSCHEN

INPUT "FAHRZEIT:         ";T
INPUT "LOKNUMMER:        ";L
INPUT "GESCHWINDIGKEIT: ";V

CALL LOK (L,V) :REM LOK STARTEN

TS=TIMER
START:
TA=TIMER
IF (TS+T)>=TA THEN GOTO START

CALL LOK (L,0): REM LOK STOPPEN
CLOSE 1 :       REM I/O KANAL SCHLIESSEN
END
```

Beim ersten Byte, das die Geschwindigkeit festlegt sind eine Reihe verschiedener Werte möglich. Die Tabelle faßt alle Werte zusammen:

Dezimal	Binär	Fahrstufe	Zusatzfunktion
0	00000	0	aus
1	00001	1	aus
2	00010	2	aus
3	00011	3	aus
4	00100	4	aus
5	00101	5	aus
6	00110	6	aus
7	00111	7	aus
8	01000	8	aus
9	01001	9	aus
10	01010	10	aus
11	01011	11	aus
12	01100	12	aus
13	01101	13	aus
14	01110	14	aus
15	01111	umschalten	aus
16	10001	0	ein
17	10001	1	ein
18	10010	2	ein
19	10011	3	ein

20	10100	4	ein
21	10101	5	ein
22	10110	6	ein
23	10111	7	ein
24	11000	8	ein
25	11001	9	ein
26	11010	10	ein
27	11011	11	ein
28	11100	12	ein
29	11101	13	ein
30	11110	14	ein
31	11111	umschalten	ein

Wichtig dabei sind die Stufen 0 (Stand), 15 (umschalten), 16 (Stand mit Zusatzfunktion ein) und 31 (umschalten mit Zusatzfunktion ein). Sofern Sie Konstanten in Ihrem Programm verwenden ist diese Tabelle sicher nützlich.

Zusatzeinrichtung

Die Zusatzfunktion (Licht, Telexkupplung oder Rauchpatrone) werden beim Märklin Digital H0 System gesondert angesteuert. Da Bits von der niederwertigsten Position (1 = 2 0) nach vorne, beginnend mit '0', gezählt werden, hat das Bit für die Zusatzfunktion die Nummer 4, was dem dezimalen Wert 16 entspricht. Wenn also eine Fahrstufe mit eingeschalteter Zusatzfunktion gefahren werden soll, ist zu der gewünschten Geschwindigkeit jeweils 16 zu addieren. Wenn Sie also bei einer Lok, wo das Licht als Sonderfunktion geschaltet wird, mit eingeschalteter Beleuchtung in Fahrtrichtung in Fahrstufe fünf fahren wollen, ist bei Geschwindigkeit eine '21'einzugeben.

Um Lokomotiven sanft beschleunigen und bremsen zu können, erweitern wir das Programm:

```
SUB INIT :REM Interface initialisieren
        OPEN "COM1: 2400, N,8,2,CS10000, DS" AS #1
END SUB
SUB LOK (LN,VS):REM Lokomotive steuern
        PRINT #1,CHR$(VS);CHR$(LN);
END SUB
SUB HALT (LN):REM Lokomotive stoppen
        PRINT #1,CHR$(0);CHR$(LN);
END SUB
SUB ANFAHR (LN,V,B)
A=0
IF V>=16 THEN A=16
FOR I=A TO V
CALL LOK (LN,I)
DELAY B:REM NACH EINER SEKUNDE WIRD V ERHÜHT
NEXT I
END SUB
SUB BREMSEN (LN,V,B)
A=0
IF V>=16 THEN A=16
FOR I=V TO A STEP -1
CALL LOK (LN,I)
DELAY B:REM NACH EINER SEKUNDE WIRD V ERHÜHT
NEXT I
END SUB
SUB ZEIT (T)
TS=TIMER
START:
TA=TIMER
IF (TS+T)>=TA THEN GOTO START
END SUB

REM Steuerprogramm für eine Lokomotive
```

```
CLS:REM BILDSCHIRM LÜSCHEN
CALL INIT :       REM INTERFACE INITIALISIEREN

INPUT "LOKNUMMER        :";LN
INPUT "ZUSATZEINR.      :";Z
INPUT "GESCHWINDIGKEIT :";V:IF Z=1 THEN V=16+V
INPUT "BESCHLEUNIGUNG  :";B
CALL ANFAHR (LN,V,B)
fahr:
IF INSTAT THEN CALL HALT(LN):stop
goto fahr

CLOSE 1 :         REM I/O KANAL SCHLIESSEN
END
```

Sortieren in Sekundenschnelle

Wenn in eigenen Programmen Daten verarbeitet werden sollen, gleichgültig, ob es numerische oder alphabetische Zeichen sind, stellt sich oft die Aufgabe, diese Daten zu ordnen. Das Ordnen von Daten oder das Sortieren ist ein Problem, das schon viele Programmierer beschäftigte. Es ist nicht zuletzt deshalb ein Grund zum Nachdenken, denn wenn man Daten mit einem Rechner sortiert, wie das ein Kartenspieler in der Hand macht, dann kann es passieren, daß man recht lange warten muß, bis der Rechner die Ordnung vollzogen hat. Hier sollen einige Ordnungsalgorithmen vorgestellt werden.

Ziel dieser Zusammenstellung ist es, verschiedene Sortiermethoden vorzustellen, sie zu vergleichen und in Modellbewegungen umzusatzen. Vielleicht ist dies aber auch nur eine Anregung, um aus den benutzten Elementen Ihre individuell beste Sortiermethode zu entwickeln.

Beschrieben werden soll das Sortieren von Zahlen, die in einem Vektor vorliegen und im Speicher des Rechners Platz finden. Diese Darstellung wird deshalb gewählt, weil es dem Rechner grundsätzlich gleichgültig ist, ob er Zahlen oder Strings vergleicht, es müßte also beim Programmieren in BASIC lediglich A(I) durch A$(I) ersetzt werden. Zahlendarstellung wurde vor allem deshalb gewählt, weil sich hier leicht und systematisch "Unordnung" herstellen läßt, die geordnet wird und deshalb einen leichten, reproduzierbaren Vergleich zwischen den einzelnen Sortiermethoden zuläßt.

Wie die Kartenspieler
Die einfachste Methode des Sortierens ist Sortieren durch direktes Einfügen. Diese Methode ist dieselbe, die Kartenspieler benutzen.

Um den entsprechenden Platz zu bestimmen, muß zwischen Vergleich und Bewegung abgewechselt werden. Das Element a(i) wird mit dem Element a(i+1) verglichen, ist a(i+1) kleiner, wird es nach links verschoben. Es gibt zwei Bedingungen, die den Prozeß des Nachlinks-Wanderns unterbrechen:
1. Ein Element a(i+1) mit kleinerem Wert als a(i) wird gefunden.
2. Das linke Ende der Zielsequenz ist erreicht.

Analysieren wir diese Sortiermethode, zeigt sich: Die Zahl C(i) der Vergleiche beim i-ten Durchlauf ist höchstens i-1 und mindestens 1, und unter der Annahme, daß die Sortierzahlen gleichmäßig verteilt sind, im Mittel i/2. Die Zahl der Bewegungen M(i) (Zuweisung von Elemenen) ist c(i)+2, inklusive dem Endwert.

Das Sortieren durch Einfügen zeigt ein "natürliches" Verhalten, das heißt, das Programm stellt rasch fest, daß eine Menge von Anfang an geordnet war und braucht man am längsten, um die maximale Unordnung zu sortieren. Gleichzeitig beschreibt der Algorithmus einen stabilen Sortierprozeß, die Reihenfolge der Elemente, die "stimmt", bleibt unverändert.

```
  10  REM ***** LINEAR SORT *****

  20  FOR I=2 TO N
  30  |  A(0)=A(I)
  40  |  J=I-1
  50  |  IF
      |     A(0) >= A(J)
      |        THEN
      |           ( GOTO ) 90
  60  |  A(J+1)=A(J)
  70  |  J=J-1
  80  |  IF
      |     J<>0
      |        THEN
      |           ( GOTO ) 50
  90  |  A(J+1)=A(0)
 100  NEXT I
```

Ein Steuerprogramm in Turbo-BASIC, das den Algorithmus in Rangierfahrten umsetzt.

```
DIM A%(5),L%(5),G%(5),M(16,31),MA%(6),K%(6),P%(2,6)
L%(1)=21:L%(2)=12:L%(3)=44:L%(4)=26:L%(5)=10
G(1)=1: G(2)=2 :G(3)=3 :G(4)=4 :G(5)=5
MA(1)=3: MA(2)=3 :MA(3)=3 :MA(4)=3 :MA(5)=3 :MA(6)=4
K(1)=11:K(2)=10:K(3)=12:k(4)=13:K(5)=4 :K(6)=8
SUB Init
        OPEN "COM1: 2400, n,8,2,CS10000, DS" AS #1
        print#1,CHR$(27);"2400,n,8,2";chr$(0);
END SUB

SUB warte
        DELAY 0.1
END SUB

SUB FAHREN (LN,VS)
        FOR I=2 TO VS
                PRINT #1,CHR$(I);CHR$(LN);
                DELAY 0.5
        NEXT I

END SUB

sub halt(LN)
        print #1,chr$(0);chr$(LN);
end sub

SUB WENDE (LN)
        PRINT #1,CHR$(15);CHR$(LN);
        PRINT #1,CHR$(15);CHR$(LN);

        delay 0.1
END SUB
SUB MAGNET (AD, ST$)
        IF ST$="g" then flag=33
        IF ST$="r" then flag=34
        PRINT #1,CHR$(flag);CHR$(AD);
        delay 0.5
        PRINT #1,CHR$(32);

END SUB
SUB LOK (LN,VS)
        PRINT #1,CHR$(VS);CHR$(LN);
        PRINT #1,CHR$(VS);CHR$(LN);
```

```
call warte
END SUB

sub magaus
        print #1,chr$(32);
        delay 0.5
end sub

SUB MODUL (M)
for S=1 to 2
        PRINT#1,CHR$(M+192);
call warte
        A=ASC(INPUT$(1,#1))
        B=ASC(INPUT$(1,#1))
next S
END SUB

SUB FABISMOD (M,K,LN,VS,R(2))

VA=1
EINLESEN:

VA=VA+1:IF VA>VS THEN VA=VS
        PRINT#1,CHR$(VA);CHR$(LN);
        delay 0.25
        PRINT#1,CHR$(M+192);
        A=ASC(INPUT$(1,#1))
                FOR I=7 TO 0 STEP -1
                        P=SGN(A AND 2^I):R(8-I,M)=P
                NEXT I
        B=ASC(INPUT$(1,#1))
                FOR I=7 TO 0 STEP -1
                        P=SGN(B AND 2^I):R(16-I,M)=P
                NEXT I
        IF R(K,M)=1 then
        CALL LOK (LN,0)
        CALL LOK (LN,0)
        call warte
        exit SUB
        end if
GOTO EINLESEN
END SUB

SUB FABISMODF (M,K,LN,VS,R(2))

VA=1
EINLESN:

VA=VA+1:IF VA>VS THEN VA=VS
        PRINT#1,CHR$(VA);CHR$(LN);
delay 0.5
        PRINT#1,CHR$(M+192);
        A=ASC(INPUT$(1,#1))
                FOR I=7 TO 0 STEP -1
                        P=SGN(A AND 2^I):R(8-I,M)=P
                NEXT I
        B=ASC(INPUT$(1,#1))
                FOR I=7 TO 0 STEP -1
                        P=SGN(B AND 2^I):R(16-I,M)=P
                NEXT I
        IF R(K,M)=0 then
        CALL LOK (LN,0)
        call warte
```

```
        exit SUB
        end if
GOTO EINLESN
END SUB

SUB G1
        CALL MAGNET(14,"g")
        CALL MAGNET(11,"g")
        CALL MAGNET(13,"r")
END SUB
SUB G2
        CALL MAGNET(14,"g")
        CALL MAGNET(13,"g")
        CALL MAGNET(11,"g")
        CALL MAGNET(9,"g")
        CALL MAGNET(12,"r")
END SUB
SUB G3
        CALL MAGNET(14,"g")
        CALL MAGNET(11,"g")
        CALL MAGNET(13,"g")
        CALL MAGNET(9,"g")
        CALL MAGNET(12,"g")
END SUB
sub G4
        CALL MAGNET(14,"g")
        CALL MAGNET(11,"g")
        CALL MAGNET(13,"g")
        CALL MAGNET(12,"g")
        CALL MAGNET(9,"r")
END SUB
sub G5
        CALL MAGNET(14,"g")
        CALL MAGNET(13,"g")
        CALL MAGNET(11,"r")
END SUB

SUB R1 (L)
print "1"
        CALL MAGNET(11,"g")
        CALL MAGNET(13,"r")
        CALL MAGAUS
call FABISMOD (3,11,L,8,M())
call WENDE (L)

END SUB

SUB R2 (L)
print "2"
        CALL MAGNET(13,"g")
        CALL MAGNET(11,"g")
        CALL MAGNET(9,"g")
        CALL MAGNET(12,"r")
        CALL MAGAUS
call FABISMOD (3,10,L,8,M())
call WENDE (L)
END SUB

SUB R3 (L)
print "3"
        CALL MAGNET(11,"g")
        CALL MAGNET(13,"g")
        CALL MAGNET(9,"g")
        CALL MAGNET(12,"g")
        CALL MAGAUS
call FABISMOD (3,12,L,8,M())
call WENDE (L)
END SUB
SUB R4 (L)
print "4"

        CALL MAGNET(11,"g")
        CALL MAGNET(13,"g")
        CALL MAGNET(12,"g")
        CALL MAGNET(9,"r")
        CALL MAGAUS
call FABISMOD (3,13,L,8,M())
call WENDE (L)
END SUB
SUB R5 (L)
print "5"

        CALL MAGNET(13,"g")
        CALL MAGNET(11,"r")
        CALL MAGAUS
call FABISMOD (3,4,L,8,M())
call WENDE (L)
END SUB

call Init
for i=1 to 5
LH=L%(i)
call lok (LH,0)
next i

for j=1 to 2
for i=1 to 4
call MODUL (i)
next i
next j

cls
goto start:
PRINT "Aufstellung"
call MAGNET (14,"g")
PRINT "Lok 1"
call FABISMOD (3,1,10,7,M())
call WENDE(10)
call FABISMOD (3,16,10,4,M())
PRINT "Lok 2"

call FABISMOD (2,5,26,7,M())
call WENDE (26)
call FABISMOD (2,12,26,4,M())
PRINT "Lok 3"

call FABISMOD (2,4,44,7,M())
call WENDE (44)
call FABISMOD (2,13,44,5,M())
PRINT "Lok 4"

call FABISMOD (4,6,12,5,M())
call WENDE (12)
PRINT "Lok 5"

call FABISMOD (4,8,21,5,M())
```

Ordnen

```
call WENDE(21)

start:
cls
print "Start des Sortierlaufs"
print
print "Eingabe der Ausgangsposition"
for i=1 to 5
korr:
print "Lokomotive ";i; "Position :";:input sf%
if i>1 then
for p=1 to i-1
        if sf%=A%(p) then korr
next p
end if
        if sf%=0 then korr
        if sf%>5 then korr
A%(i)=sf%
next i
cls
print "Sortierreihenfolge Start"

for i=1 to 5
print A%(i),
next i
goto algo
for j=1 to 5
print a%(j)
LNR=L%(j)
P%(1,j)=LNR
print LNR
P%(2,j)=A%(j)
if A%(j)=1 then
        call G1
        call FABISMOD (3,11,LNR,6,M())
        call WENDE (LNR)
end if
if A%(j)=2 then
        call G2
        call FABISMOD (3,10,LNR,6,M())
        call WENDE (LNR)
end if
if A%(j)=3 then
        call G3
        call FABISMOD (3,12,LNR,6,M())
        call WENDE (LNR)
end if
if A%(j)=4 then
        call G4
        call FABISMOD (3,13,LNR,6,M())
        call WENDE (LNR)
end if
if A%(j)=5 then
        call G5
        call FABISMOD (3,4,LNR,6,M())
        call WENDE (LNR)
end if

next j
for i=1 to 5
LH=L%(i)
call halt (LH)
next i

algo:
REM Algorithmus
rem Linear SORT
call MAGNET (14,"r")
for i=2 to 5
A%(0)=A%(i)
LV=L%(A%(i))
call FABISMOD (1,14,LV,6,M())
call warte
call WENDE(LV)
j=i-1
vergleich:
print "Vergleich"
if A%(0) >= A%(j) then tausche
A%(j+1)=A%(j)
L1=L%(j+1)
L2=L%(j)
call FABISMOD (2,3,L2,6,M())
call warte
call WENDE(L2)
print j
SELECT CASE (j+1)
     case 1
     call R1 (L2)
     case 2
     call R2 (L2)
     case 3
     call R3 (L2)
     case 4
     call R4 (L2)
     case 5
     call R5 (L2)
end select
for k=1 to 5
LH=L%(i)
call halt (LH)
next k
print j
SELECT CASE (j)
     case 1
     call R1 (LV)
     case 2
     call R2 (LV)
     case 3
     call R3 (LV)
     case 4
     call R4 (LV)
     case 5
     call R5 (LV)
end select
for k=1 to 5
LH=L%(i)
call halt (LH)
next k

j=j-1
if j<>0 then vergleich
tausche:
print "Tausche"
A%(j+1)=A%(0)
print j
SELECT CASE j+1
```

```
        case 1
        call R1 (LV)
        case 2
        call R2 (LV)
        case 3
        call R3 (LV)
        case 4
        call R4 (LV)
        case 5
        call R5 (LV)

end select

for k=1 to 5
LH=L$(i)
call halt (LH)
next k

for k=1 to 5:print A%(k),:next k

next i
```

Sind Felder zu sortieren, bei denen lediglich wenige Elemente am falschen Platz stehen, empfiehlt sich als Modifikation des Linear-Sorts der Ripple-Sort. Diese Methode stellt bereits beim ersten Durchlauf fest, ob es sich überhaupt lohnt, die Sortierarbeit zu tun. Sie ist die schnellste Methode für den trivialen Fall, das heißt, wenn die Daten bereits sortiert vorliegen.

```
   10  REM ***** RIPPLE SORT *****

   20  C=0
   30  FOR J=1 TO N-1
   40  |  IF
       |     A(J) <= A(J+1)
       |        THEN
       |           ( GOTO ) 60
   50  |  A(0)=A(J):
       |  A(J)=A(J+1):
       |  A(J+1)=A(0):
       |  C=1
   60  NEXT J
   70  IF
          C=1
             THEN
                ( GOTO ) 20
```

Das Gegenteil

Das Sortieren durch Einfügen scheint keine ideale Methode für DV-Anlagen zu sein, da beim Einfügen eines Elementes ganze "Zeilenreste" verschoben werden. Bessere Ergebnisse sollte eine Methode bringen, die einzelne Elemente über größere Entfernungen verschiebt. Das Sortieren durch Auswählen, das diese Idee verfolgt, funktioniert durch

- Auswahl des kleinsten Elements
- Austausch gegen das erste Element.

Daraufhin wiederholt man dieses Verfahren, bis letztlich das größte Element übrigbleibt. Die direkte Auswahl ist offensichtlich das Gegenteil vom direkten Einfügen. Direktes Einfügen zieht nur das nächste Element des Ausgangsfeldes in Betracht, während direkte Auswahl alle Elemente des Zielfeldes betrachtet, um das kleinste Element zu finden.

```
   10  REM ***** AUSWAHL SORT *****

   20  I=0:
       J=0:
       M=0
```

```
 30  FOR I=1 TO N-1
 40  |  M=I
 50  |  FOR J=I+1 TO N
 60  |  |  IF
     |  |     A(J) < A(M)
     |  |        THEN
     |  |           M=J
 70  |  NEXT J
 80  |  A(0)=A(I):
     |  A(I)=A(M):
     |  A(M)=A(0)
 90  NEXT I
```

Vergleiche und tausche

Als dritte und letzte einfache Sortiermethode soll das Sortieren durch direktes Austauschen vorgestellt werden. Der Algorithmus beruht auf dem Prinzip des fortgesetzten Vergleichens und Austauschens von benachbarten Elementen, bis alle sortiert sind. Wie bei der direkten Auswahl wird das Fehld mehrmals durchlaufen. Jedesmal wandert das kleinste Element zum linken Ende des Feldes. Stellt man sich das Feld nun vertikal statt horizontal vor, so steigt bei jedem Durchlauf ein Element als "Blase" (Bubble) auf den seiner Größe entsprechenden Platz. Daher hat die Methode ihren Namen: Bubble-Sort.

```
 10  REM ***** BUBBLE SORT *****

 20  FOR I=2 TO N
 30  |  FOR J=N TO I STEP -1
 40  |  |  IF
     |  |     A(J-1) > A(J)
     |  |        THEN
     |  |           ( GOTO ) 60
 50  |  |  GOTO 70
 60  |  |  A(0)=A(J):
     |  |  A(J)=A(J-1):
     |  |  A(J-1)=A(0)
 70  |  NEXT J
 80  NEXT I
```

Dieser Algorithmus läßt sich verbessern, wenn man sich merkt, ob während eines Durchlaufs überhaupt ein Austausch, also eine Umordnung der Elemente stattgefunden hat. Dies läßt sich noch weiter optimieren, wenn man sich merkt, in welcher Position der letzte Austausch stattfand. Darüber hinaus kann man auch noch die Richtungen der aufeinanderfolgenden Durchgänge ändern. Dies führt zum Shake-Short, ein Bubble-Sort, bei dem die aufsteigenden Blasen durcheinandergeschüttelt werden.

```
 10  REM ***** SHAKE SORT *****

 20  J=2:
     R=N:
     K=N
 30  FOR J=R TO L STEP -1
 40  |  IF
     |     A(J-1) <= A(J)
     |        THEN
     |           ( GOTO ) 70
 50  |  A(0)=A(J-1):
     |  A(J-1)=A(J):
     |  A(J)=A(0)
 60  |  K=J
 70  NEXT J
 80  L=K+1
 90  FOR J=L TO R
100  |  IF
     |     A(J-1) <= A(J)
```

```
      |        THEN
      |           ( GOTO ) 130
  110 |  A(0)=A(J-1):
      |  A(J-1)=A(J):
      |  A(J)=A(0)
  120 |  K=J
  130 NEXT J
  140 R=K+1
  150 IF
         L <= R
            THEN
               ( GOTO ) 30
```

Nun wollen wir uns drei verbesserte Versionen der elementaren Sortiermethoden genauer ansehen. Alle direkten Methoden bewegen im wesentlichen bei jedem Schritt jedes Element. Jede Verbesserung muß also auf dem Prinzip beruhen, Elemente über größere Distanzen, also über die direkte Nachbarschaft hinaus, bei einzelnen Schritten zu bewegen.

Blasen steigen auf
1950 schlug D.L.Shell eine Verbesserung des Sortierens durch direktes Einfügen vor. Die Sortiermethode sieht aus:
Bei einem Array mit N Elemente werden Paare aus den Elementen A(i) und A(i+(N/2)) gebildet. Aus diesen werden Elemente zusammengefaßt, so daß sich N/4 Paare ergeben. Dies wird nun so lange fortgeführt, zusammenfassen, ordnen, zusammenfassen, ordnen..., bis eine Liste mit N Elementen übrigbleibt, das Resultat.

```
   10 REM ***** SHELL SORT *****

   20 M=N
   30 M=INT(M/2)
   40 IF
         M=0
            THEN
               ( GOTO ) 170
   50 J=1
   60 K=N-M
   70 I=J
   80 L=I+M
   90 IF
         A(I) < A(L)
            THEN
               ( GOTO ) 140
  100 A(0)=A(I):
      A(I)=A(L):
      A(L)=A(0)
  110 I=I-M
  120 IF
         I < 1
            THEN
               ( GOTO ) 140
  130 GOTO 80
  140 J=J+1
  150 IF
         J<=K
            THEN
               ( GOTO ) 70
  160 GOTO 30
```

Das Sortieren durch direkte Auswahl beruht aus der fortgesetzten Auswahl des jeweils kleinsten Elements. Das Sortieren durch Auswahl läßt sich nur verbessern, wenn in jedem Schritt mehr Informationen gewonnen werden kann als die Erkenntnis, welches das jeweils kleinste Element ist.

Beispielsweise kann mit n/2-Vergleichen der kleinere von jedem Paar von Elementen gefunden werden. Setzt man dies fort, so läßt sich mit n-1-Schritten ein Auswahlbaum aufbauen und die Wurzel als kleinstes Element bestimmen. Der zweite Schritt ist das Begehen des durch den kleinsten Schlüssel markierten Wegs. Dabei ersetzt man das kleinste Element durch das Element des anderen Zweiges bei den Knoten. Dieser Baumspaziergang kann wiederholt werden, bis der Baum leer ist, und das Sortieren ist beendet. Allerdings ist dieses Verfahren aufwendig, da eine Buchführung über den Wanderweg nötig ist.

Ein "Heap"-Baum
Eine übersichtliche Organisation und eine geordnete Buchführung des Sortierens wird bei einem Verfahren erreicht, das J.Williams 1964 vorstellte und dem er den Namen Heap-Sort gab. Ein Heap (= Haufen) ist definiert als eine Folge von Schlüsseln h(1), h(1+1), ..., h(r), die der Bedingung
h(i) = h(2*i)
h(i) = h(2*i+1) für alle i = 1...r/2
genügen. Wenden wir die Definition auf den Beispielsfall an, so ist einsichtig, daß die Bäume "Heaps" sind und daß das kleinste Element an der Spitze steht.

```
   10  REM ***** HEAP SORT *****

   20  I=N-1:
       L=2:
       R=N
   30  FOR J=R TO L STEP -1
   40  |  IF
       |     A(J-1)<=A(J)
       |        THEN
       |           ( GOTO ) 70
   50  |  A(0)=A(J):
       |  A(J)=A(J-1):
       |  A(J-1)=A(0)
   60  |  I=J
   70  NEXT J
   80  L=I+1
   90  FOR J=L TO R
  100  |  IF
       |     A(J-1)<=A(J)
       |        THEN
       |           ( GOTO ) 130
  110  |  A(0)=A(J):
       |  A(J)=A(J-1):
       |  A(J-1)=A(0)
  120  |  I=J
  130  NEXT J
  140  R=I-1
  150  IF
          L<=R
             THEN
                ( GOTO ) 30
```

Blasen "quick"-gelenkt
Als letzte und schnellste Sortiermethode sei ein Verfahren vorgestellt, das Einfügen und Auswählen vereint und auf dem Prinzip des Austauschens basiert. Obwohl diese Methode auf dem Bubble-Sort aufbaut, der im Vergleich nicht so überzeugt, bringt die Verbesserung des Bubble-Sort, vorgeschlagen von C.A.R.Hoare, der sie als Quick-Sort bezeichnet, die besten Ergebnisse. Ein Prinzip von Quick-Sort ist der Austausch über große Distanzen. Liegen zum Beispiel Elemente in umgekehrter Reihenfolge ihrer Wertigkeit vor, kann man sie mit nur n/2-Austauschvorgängen sortieren, wenn man den Array von den beiden Enden, zur Mitte fortschreitend, vertauscht.

```
 10  REM **** QUICK SORT *****

 20  L=1
 30  B(L)=N+1
 40  M=1
 50  J=B(L)
 60  I=M-1
 70  IF
        J-M < 3
           THEN
              ( GOTO ) 230
 80  M1=INT((I+J)/2)
 90  I=I+1
100  IF
        I=J
           THEN
              ( GOTO ) 170
110  IF
        A(I)<=A(M1)
           THEN
              ( GOTO ) 90
120  J=J-1
130  IF
        I=J
           THEN
              ( GOTO ) 170
140  IF
        A(J)>= A(M1)
           THEN
              ( GOTO ) 120
150  A(0)=A(I):
     A(I)=A(J):
     A(J)=A(0)
160  GOTO 90
170  IF
        I >= M1
           THEN
              I=I-1
180  IF
        J=M1
           THEN
              ( GOTO ) 200
190  A(0)=A(I):
     A(I)=A(M1):
     A(M1)=A(0)
200  L=L+1
210  B(L)=I
220  GOTO 50
230  IF
        J-M < 2
           THEN
              ( GOTO ) 260
240  IF
        A(M) < A(M+1)
           THEN
              ( GOTO ) 260
250  A(0)=A(M):
     A(M)=A(M+1):
     A(M+1)=A(0)
260  M=B(L)+1
270  L=L-1
280  IF
        L>0
```

```
            THEN
               ( GOTO ) 50
```

Diese Methode läßt sich ausbauen. Zunächst wählt man willkürlich ein Element aus, durchsuche dann den Array von links, bis ein größeres Element gefunden ist, und von rechts, bis ein kleineres Element gefunden wird. Nun werden die Elemente vertauscht, und der Suchlauf beginnt neu. Setzt man dies lang genug fort, so treffen sich die Suchläufe in der Mitte des Arrays. Der Schlüssel zu dem schrittweisen Zerlegen des Arrays liegt in der Buchführung.

Man muß sich merken, welche Zerlegung noch nicht ausgeführt ist, denn zunächst kann ja nur ein Suchlauf durchgeführt werden, entweder von links oder von rechts. Insgesamt eignet sich Quick-Sort zum Sortieren großer Felder, allerdings gibt es einige pathologische Fälle, wo Quick-Sort recht langsam wird. Am wirkungsvollsten arbeitet Quick-Sort bei größtmöglicher Unordnung.

Grundsätzlich, das zeigen Untersuchungen von N. Wirth, ist Quick-Sort die schnellste Methode. Dabei ist zu beachten, daß der Vorsprung von Quick-Sort noch größer wird, wenn nicht nach einfachen Zahlen sondern nach Schlüssel geordnet wird. Dies leuchtet ein, da mit Quick-Sort bei echten Sortieraufgaben am wenigsten umgestellt werden muß, was natürlich zusätzlich Rechenzeit spart. Allerdings ist Ripple-Sort die intelligenteste Methode, sie merkt am schnellsten, ob etwas zu ordnen ist.

Deshalb bietet sich für routinemäßiges Sortieren an, zunächst mit einer kleinen Vorroutine festzustellen, ob sich der Sortierjob lohnt. Mit einer Abfrage, ob sortieren nötig wird, kann dann der Sprung in eine der schnellen Sortierroutinen erfolgen. Der Phantasie zum effektiven Programmieren sind keine Grenzen gesetzt.

Sortieren am Ablaufberg

Die einfachste Sortierung, die sich mit der Digital-HO-Bahn simulieren läßt, ist das Sortieren am Ablaufberg. Diese Art zu sortieren entspricht der Methode, mit der ein Kartenspieler seine Karten sortiert. Im Dialog wird die Reihenfolge der Wagen der Zuggarnitur eingegeben. Programmgesteuert werden die Wagen auf der Gleisharfe in aufsteigender Reihenfolge abgestellt. Das Programm wurde für Turbo BASIC erstellt und arbeitet mit Prozeduren.

```
REM Globale Variable
DIM R(16,31)      :REM 31 Module mit je 16 Kontakten
DIM LSTAT(80,2)   :REM 80 Lokomotiven, vmin, vaktuell
DIM MSTAT(255)    :REM Stellung der 255 Magnetartikel (g=0,r=1)

REM basic procedures
SUB warte : REM Pause, um eine sichere Schnittstellenbedienung zu erreichen
        for T=1 to 50:next t
END SUB

SUB Init : REM Die Schnittstelle wird initialisiert
        OPEN "COM1: 2400, n,8,2,CS10000, DS" AS #1
END SUB

SUB LOK1 (LN,VS): REM zum Anfahren wird die Lok beschleunigt
        PRINT #1,CHR$(VS+2);CHR$(LN);
call warte
        PRINT #1,CHR$(VS);CHR$(LN);
        PRINT #1,CHR$(VS);CHR$(LN);
END SUB

SUB LOK (LN,VS): REM Normalfahrt der LOK
        PRINT #1,CHR$(VS);CHR$(LN);
        PRINT #1,CHR$(VS);CHR$(LN);
```

```
END SUB

SUB MAGNET (AD, ST$): REM Die Weiche wird gestellt
        IF ST$="g" then flag=33
        IF ST$="r" then flag=34
        PRINT #1,CHR$(flag);CHR$(AD);
call warte
END SUB

SUB MAGAUS: REM der Magnetartikel wird abgeschletet
            REM Für Entkupplungsgleise notwendig
        PRINT #1, CHR$(32);
        PRINT #1, CHR$(32);
END SUB

SUB RMMOD (M,K,L,R(2)): REM Die Lok fährt bis zum Rückmeldekontakt
EINLESEN:
        PRINT#1,CHR$(M+192);
call warte
        A=ASC(INPUT$(1,#1))
                FOR I=7 TO 0 STEP -1
                        P=SGN(A AND 2^I):R(8-I,M)=P
                NEXT I
        B=ASC(INPUT$(1,#1))
                FOR I=7 TO 0 STEP -1
                        P=SGN(B AND 2^I):R(16-I,M)=P
                NEXT I
        IF R(K,M)=1 then CALL LOK (L,0):exit SUB
GOTO EINLESEN
END SUB

SUB RMCMOD (M): REM Diese Prozedur wird benutzt um Module ohne Auswertung einzulesen,
                    um einen kontrollierten Zustand der Dekoder zu erreichen
for S=1 to 2
        PRINT#1,CHR$(M+192);
call warte
        A=ASC(INPUT$(1,#1))
        B=ASC(INPUT$(1,#1))
next S
END SUB

SUB RMMOD1 (M,K,L,R(2)): REM Die Lokomotive fährt,
                             bis der Rückmeldekontakt wieder frei ist
EIN:
        PRINT#1,CHR$(M+192);
call warte
        A=ASC(INPUT$(1,#1))
                FOR I=7 TO 0 STEP -1
                        P=SGN(A AND 2^I):R(8-I,M)=P
                NEXT I
        B=ASC(INPUT$(1,#1))

                FOR I=7 TO 0 STEP -1
                        P=SGN(B AND 2^I):R(16-I,M)=P
                NEXT I
        IF R(K,M)=0 then CALL LOK (L,0):exit SUB

GOTO EIN
END SUB

SUB AB (L): REM Ein Wagen wird abgekuppelt
        CALL LOK1 (L,8)
call warte
```

Ordnen

```
FOR I=1 to 15
        CALL MAGNET (16,"r")
call warte
NEXT I
        CALL MAGAUS
call warte
        CALL LOK (L,0)

END SUB

SUB RMMODK (M,K,R(2)): REM ein Rückmeldekontakt wird abgefragt und ausgewertet,
                          um festzustellen, ob ein Wagen angekommen ist
EINLES:
        PRINT#1,CHR$(M+192);
call warte
        A=ASC(INPUT$(1,#1))
                FOR I=7 TO 0 STEP -1
                        P=SGN(A AND 2^I):R(8-I,M)=P
                NEXT I
        B=ASC(INPUT$(1,#1))
                FOR I=7 TO 0 STEP -1
                        P=SGN(B AND 2^I):R(16-I,M)=P
                NEXT I
        IF R(K,M)=1 then exit SUB
GOTO EINLES
END SUB

MAIN:
        CALL Init
        CLS
        PRINT" LOK (L) fährt mit FAHRSTUFE (v) bis MODUL (M) KONTAKT (K) "
        INPUT "LOKNUMMER"; L
        INPUT "FAHRSTUFE"; V
        INPUT "MODUL";M
        INPUT "KONTAKT";K
        PRINT "Sortierreihenfolge"
        for z=1 to 5
        INPUT S%(z)
        next z
REM Der Zug fährt bis zum Entkupplungsgleis
FOR Z=1 to 5
        CALL RMCMOD(M)
        CALL LOK (L,V)
        CALL LOK (L,V)
CALL RMMOD (M,K,L,R()):REM Fahrfehler werden ausgeglichen
        CALL LOK (L,15)
        CALL LOK (L,15)
REM Es wird entschieden auf welches Gleis der Wagon rollen soll
SELECT  CASE S%(z)
        CASE 1
        GOSUB 1
        CASE 2
        GOSUB 2
        CASE 3
        GOSUB 3
        CASE 4
        GOSUB 4
        CASE 5
        GOSUB 5
END SELECT
        CALL AB(L): REM der Wagen wird abgekuppelt
        CALL LOK (L,15)
        CALL LOK (L,15)
```

```
        CALL RMMODK (4,8,R()): REM die Lok schiebt den ZUg bis zum Entkupplunggleis
NEXT Z
        STOP

1:
REM Fahrstraße 1
        CALL MAGNET(14,"g")
        CALL MAGNET(11,"g")
        CALL MAGNET(13,"r")
        CALL MAGAUS
RETURN
2:
REM Fahrstraße 2
        CALL MAGNET(14,"g")
        CALL MAGNET(13,"g")
        CALL MAGNET(11,"g")
        CALL MAGNET(9,"g")
        CALL MAGNET(12,"r")
        CALL MAGAUS

RETURN
3:
REM Fahrstraße 3
        CALL MAGNET(14,"g")
        CALL MAGNET(11,"g")
        CALL MAGNET(13,"g")
        CALL MAGNET(9,"g")
        CALL MAGNET(12,"g")
        CALL MAGAUS

RETURN
4:
        CALL MAGNET(14,"g")
        CALL MAGNET(11,"g")
        CALL MAGNET(13,"g")
        CALL MAGNET(12,"g")
        CALL MAGNET(9,"r")
        CALL MAGAUS
RETURN
5:
REM Fahrstraße 5
        CALL MAGNET(14,"g")
        CALL MAGNET(13,"g")
        CALL MAGNET(11,"r")
        CALL MAGAUS
RETURN
```

Turbo-Prolog stellt die Weichen

Eine Sprache der künstlichen Intelligenz zur Steuerung einer Bahnanlage

Programming in Logic (Prolog) ist sicher genau die richtige Sprachumgebung, um Steuerungsaufgaben für Eisenbahnlagen zu realisieren. Die Grundstrukturen der Sprache ermöglichen es, komplexe Situationen, die sich aus logischen Verknüpfungen vieler Einzelbedingungen ergeben, auf einfache Weise zu analysieren. Gerade diese Problemstellungen treten bei der Erstellung von Steuerungsprogrammen für Bahnanlagen immer wieder auf, so daß es nahe liegt, Prolog zu deren Lösung einzusetzen. Wie das geschehen kann, und welche Grundsätze zu beachten sind, das soll im folgenden am Beispiel des MÄRKLIN Digital-Systems beschrieben werden.

Die Ansteuerung der seriellen Schnittstelle

Obwohl Prolog nicht für die maschinennahe Programmierung konzipiert wurde, bietet Turbo-Prolog hierfür einige Hilfen. Die eleganteste Lösung wäre die Einbindung von Assembler- oder C-Routinen zur Bedienung des seriellen Ports. Da in diesem Fall jedoch keine großen Anforderungen an Übertragungsgeschwindigkeit und Bedienung der Steuerleitungen (Handshake) gestellt werden, ist es mit Hilfe der "Low Level Unterstützung" durch das Turbo-System möglich, die Treiberroutinen ebenfalls in Prolog zu realisieren.

Das Interface des MÄRKLIN Digital-Systems bedient neben den Datenleitungen lediglich die CTS-Leitung der V 24 Schnittstelle, so daß der Zugriff auf BIOS-Treiberroutinen des Computers Probleme bereiten könnte. Um diesbezüglich allen Schwierigkeiten aus dem Weg zu gehen, erfolgt die Ansteuerung der Schnittstelle über den Portbaustein INS 8250 direkt. Der INS 8250 besitzt 9 Register, deren Belegung die Arbeitsweise des Bausteins beeinflußt, und über deren Inhalte Informationen über den Stand der seriellen Kommunikation gewonnen werden können. Jedes Register ist über einen Ein/Ausgabe-Kanal (E/A-Kanal) des Computers erreichbar, so daß sie ausgelesen bzw. beschrieben werden können. Abbildung 1 zeigt die Zuordnung zu den Adressen der einzelnen E/A-Kanäle. Dabei muß beachtet werden, daß den E/A-Adressen $3f8 ($2f8) bzw. $3f9 ($2f9) jeweils zwei Register zugeordnet sind. Welches dieser beiden beim Zugriff auf den entsprechenden Kanal angesprochen wird, hängt von Bit 7 des 4. Registers ab. Ist es auf 1 gesetzt, dann wird jeweils das b-Register erreicht, sonst das a-Register. Die b-Register werden jedoch nur für die Initialisierung der Schnittstelle benötigt, so daß im normalen Betrieb Bit 7 von Register 4 auf 0 gesetzt werden muß.

MÄRKLIN Digital arbeitet mit einem Übertragungsprotokoll von 2400 Baud, 8 Datenbits, keine Parität und zwei Stopbits. Um die Schnittstelle auf dieses Protokoll einzustellen, müssen die Register mit den Werten aus Tabelle 1 geladen werden. Das geschieht über die E/A-Kanäle des Computers mit Hilfe des Standardprädikats port_byte(Kanalnummer, Byte). Dieses Prädikat ermöglicht dem Prolog-Progammierer den direkten Zugriff auf die E/A-Kanäle, so daß die Initialisierung des INS 8250 vorgenommen werden kann. Durch Kanalnummer wird ein E/A-Kanal ausgewählt, die Variable muß deshalb bei Aufruf von port_byte gebunden sein. Byte kann sowohl gebunden als auch frei sein. Ist sie gebunden, dann wird dieser Wert über den ausgewählten Kanal ausgegeben. Andernfalls wird ein Zeichen über den Kanal eingelesen und an Byte gebunden, so daß nach Rückkehr von port_byte das eingelesene Zeichen zur Verfügung steht.

Prädikate zur Bedienung der seriellen Schnittstelle

Das Programm enthält eine Reihe von Prädikaten, die die Kommunikation über die serielle Schnittstelle vereinfachen und sozusagen die "Treiber" darstellen. Die gewählten Kanaladressen beziehen sich auf COM1: und müssen für die Verwendung von COM2: entsprechend geändert werden. Dem Prädikat init_v24 kann entnommen werden, wie port_byte eingesetzt wird. Beachten Sie die Belegung des Registers 4, um 1b bzw. 2b zu erreichen.

Der serielle Datenaustausch mit dem Digital-Interface erfolgt über Register 1 (DATA-Register) des INS 8250. Wird ein Wert in dieses Register geschrieben, dann gibt ihn der Baustein seriell an das Interface aus. Empfangene Daten können aus dem DATA-Register eingelesen werden. Es ist zu beachten, daß sowohl der INS 8250 als auch das Interface selbst, Zeit für die Verarbeitung von Daten benötigen und nicht immer sende- bzw. empfangsbereit sind. Das Interface benutzt die CTS-Leitung der V 24-Schnittstelle, um anzuzeigen, daß Daten übermittelt werden können. Bevor ein Datum ausgegeben wird, muß deshalb der Pegel dieser Leitung überprüft werden. Das Prädikat cts übernimmt diese Aufgabe. Es ist nur dann erfüllt, wenn die CTS-Leitung aktiv ist, das Interface also Kommunikationsbereitschaft signalisiert. Um CTS zu prüfen wird Register 7 (MODEM-Status) eingelesen und ausgewertet. Jedes Bit dieses Register enthält Informationen über die Pegel an den Steuerleitungen der seriellen Schnittstelle. Abbildung 2 zeigt die Zuordnung zwischen den einzelnen Bits dieses Registers und den Schnittstellenleitungen.

Bei den verwendeten Übertragungsparametern werden mit jedem Datenbyte 11 Bits übertragen. Das dauert bei einer Übertragungsrate von 2 400 Baud ca. 4,6 ms. Da der Computer wesentlich schneller arbeiten kann, muß vor Ausgabe eines Bytes an das DATA-Register des INS 8250 geprüft werden, ob es schon wieder frei ist, sonst könnte es zu

^[berschreibungen kommen. Das Prädikat spl (Sendepuffer leer) übernimmt diese Aufgabe. Es ist nur dann erfüllt, wenn das DATA-Register neue Sendedaten aufnehmen kann. Informationen hierüber werden anhand des Registers 6 (LINE-Status, Abb.: 2) gewonnen. Bit 6 dieses Registers zeigt an, ob der Ausgabepuffer leer ist. Es wird durch spl ausgewertet.

LINE-Status enthält auch Informationen darüber, ob ein Zeichen von der Schnittstellen empfangen wurde. Bit 0 dieses Registers ist auf 1 gesetzt, wenn ein Datenbyte angekommen ist. epv (Empfangspuffer voll) macht davon Gebrauch. Das Prädikat ist nur dann erfüllt, wenn ein Zeichen im DATA-Register bereitliegt.

Die Ein- bzw. Ausgabe über die serielle Schnittstelle wird für den Benutzer durch die Prädikate s_read(Wert) und s_write(Wert) erleichtert. Wird s_write aufgerufen muß Wert an eine Integerzahl gebunden sein. Es wird vermittels warte_auf_ausgabe so lange gewartet, bis INS 8250 und Digital-Interface empfangsbereit sind, erst dann wird Wert an die Schnittstelle übergeben. warte_auf_ausgabe nutzt die Prädiakte cts und epv. Mit Hilfe von s_read, können Zeichen von der Schnittstelle eingelesen werden. warte_auf_eingabe bewirkt, daß erst dann auf das DATA-Register zugegriffen wird, wenn ein Zeichen empfangen wurde. Dieses Zeichen wird dann an Wert gebunden. Wenn das Digital-Interface bei Aufruf von s_read oder s_write nicht kommunikationsbereit ist, dann führt das in eine Endlosschleife und eventuell zum Stack-Überlauf des Computers.

Mit den beschriebenen Prädikaten vereinfachen sich die Zugriffe auf die serielle Schnittstelle, so daß der Datenaustausch mit dem Digital-Interface einfach abgewickelt werden kann. Die beschriebene Art der Schnittstellenansteuerung ist jedoch nur bis 2 400 Baud möglich, da sonst Laufzeitprobleme auftreten. Bei der Verwendung des MÄRKLIN Training-Systems, das ja höhere Übertragungsgeschwindigkeiten zuläßt, sollten die Baudraten daher auch nicht höher eingestellt werden. Abbildung 3 zeigt, wie das Verbindungskabel zwischen serieller Schnittstelle und Interface ausgelegt sein muß.

Grundprädikate zur Steuerung der Bahnanlage

Zur Steuerung der Bahnanlage ist es sinnvoll, für elementare Anwendungen Hilfsmittel zu schaffen, die in komplexe Steuerprogramme eingebunden werden können, die jedoch immer wieder benötigt werden und deshalb als Programm-Module bereitstehen sollten. Hierzu gehören Prädikate mit denen die verschiedenen Elemente einer Anlage auf einfache Weise angesteuert werden können. Die im Programmlisting dargestellten Grundprädikate sollen dazu dienen und im folgenden näher erläutert werden.

warten(Zeit) dient zur Realisierung von Wartezeiten zwischen 1 und 59 Sekunden. warten macht von dem Standard-Prädikate time gebrauch, das die Systemzeit des Computers ermittelt. (Hier werden allerdings nur die Sekunden ausgewertet.) Der an Zeit gebundene Wert wird zu dem Sekundenwert der Systemuhr addiert und durch das Prädikat nicht(Neue_Zeit) die Uhr so lange abgefragt, bis die Wartezeit verstrichen ist. warten ist ein nützliches Hilfsprädikat und kann an vielen Stellen eingesetzt werden.

Mit fahrstufe(Loknummer, Stufe, Funktion), können einzelne Loks angesteuert werden. Um für die Loks einfache Kennungen zu haben, wird durch zuordnung eine Datenbasis geschaffen, in der zu den von l1, l2, l3, usw. durchnumerierten Loks die zugehörigen Adresse des Digital-System festgelegt sind. Als Loknummer wird deshalb immer der symbolische Wert l1, l2 usw. übergeben. Mit Hilfe der Datenbasis zuordnung kann die zugehörige Digital-Systemadresse ermittelt und die ausgewählte Lok angesprochen werden. Stufe ist eine Integerzahl, deren Größe der gewählten Fahrstufe entspricht. Durch Funktion kann die Magnetkupplung der Lok beeinflußt werden. Es wird für Funktion ein oder aus erwartet. Die Variablenwerte werden auf Korrektheit überprüft und fahrstufe wird nicht erfüllt, wenn falsche Werte übergeben wurden. Beachten Sie, daß die Magnetkupplung nicht dauernd eingeschaltet sein darf.

Mit fahrtrichtung(Nr) kann die Fahrtrichtung der durch Nr ausgewählten Lok geändert werden. Als Nr wird auch hier die symbolische Lok-Bezeichnung (l1, l2, ...) erwartet, aus der mit Hilfe von zuordnung die Lokadresse des Digitalsystems ermittelt wird.

Die Ansteuerung der Weichen erfolgt mittels weiche_stellen(Nr, Richtung) in gleicher Weise wie für die Loks. Durch zuordnung werden Relationen zwischen den Weichennummern w1, w2, ..., und den zugehörigen Adressen des Digital-Systems hergestellt. Bei Ansteuerung der Weichen muß an die Variable Stellung der Wert links oder rechts gebunden sein, damit weiche_stellen erfüllt wird. Dasselbe gilt für signal_stellen(Nr, Stellung). Als Nr sind hier Werte s1, s2 usw. erlaubt, an Stellung müssen die Werte rot oder grün gebunden sein. Das Digital-System verlangt, daß nach Betätigung eines elektromagnetischen Antriebes ein Abschaltbefehl gegeben wird. Dieser kann mit Hilfe von s_write ausgegeben werden.

Mit stop und go kann der Notaus- bzw. Freigabe-Befehl erteilt werden.

Um Informationen über den Status der Anlage zu erhalten, müssen die Rückmeldemodule (Decoder s88) abgefragt werden. Das kann auf zwei verschiedene Arten geschehen. Entweder es wird ein bestimmtes Modul angesprochen, oder es werden alle Module bis zu einer bestimmten Modulnummer abgefragt. Hier wird von der zweiten Möglichkeit Gebrauch gemacht. Es ist jedoch einfach, auch die erste zu realisieren, es soll dem Leser überlassen bleiben, dieses zu tun.

Die Abfrage der ersten n Module erfolgt, indem an das Digital-Interface der Wert 128 + n ausgegeben wird. Unmittelbar darauf sendet das Interface für jedes Rückmeldemodul zwei Bytes an den Computer, denen Informationen über den Status der Bahnanlage entnommen werden können. Um die ersten 5 Module abzufragen muß demnach der Wert 133 (128 + 5) an das Interface gesendet werden, daraufhin übermittelt es 10 Bytes, für jedes Rückmeldemodul 2.

Das Prädikat hole_status(Nr, Werteliste) erlaubt die Abfrage aller Rückmeldemodule bis zu dem durch Nr festgelegten Modul. An Werteliste werden die eingelesenen Statusworte gebunden. Es handelt sich dabei um eine Liste von Integerzahlen, deren Reihenfolge der Anordnung der Module entspricht. Erster Wert erstes Modul, zweiter Wert zweites Modul usw. Da das Interface byteweise übermittelt, wird durch erstelle_liste zunächst eine Hilfsliste einzelner Bytes eingelesen. umwandeln wandelt die so ermittelte Hilfsliste in die Werteliste um, in der jedem Rückmeldemodul genau eine Integerzahl zugeordnet ist. Wie die Auswertung der einzelnen Statusworte erfolgen kann, kann in dem Steuerprogrammbeispiel verfolgt werden.

Diese Grundprädikate ermöglichen die Ausführung komplexer Steuerungen, die nahezu allen Anforderungen gerecht werden. Die individuelle Auslegung der Datenbasis zuordnung erlaubt es, Kennungen und Zuordnungen den eigenen Bedürfnissen anzupassen, so daß übersichtliche Programmierung möglich ist. Wie mit Hilfe dynamischer Datenbanken Steuerungen realisiert werden können, soll in dem folgenden Beispiel beschrieben werden.

<u>Beispiel eines Steuerungsprogramms</u>

Die Abbildung zeigt die Auslegung einer sehr einfachen Anlage, für die ein Steuerprogramm geschrieben werden soll. Sie enthält zwei Weichen w1 und w2, vier Signale s1 bis s4 und zwei Loks l1 und l2. Über ein Rückmeldemodul können zwei Schaltgleise (A und B) und mittels Kontaktgleisstücken die Belegung der Gleise G1 und G2 abgefragt werden. Die Tabelle gibt Aufschluß über die Beschaltung der einzelnen Kontakte des Decoders s 88. Dadurch kann ermittelt werden, welche Bits des zurückgemeldeten Statuswortes den entsprechenden Kontakten des Rückmeldemoduls zugeordnet sind. Die Computersteuerung soll lediglich die Signale und Weichen einbeziehen, beide Loks werden von Hand gesteuert. Es wird allerdings durch die Signale mit Hilfe der Fahrstromunterbrechung indirekt auf die Steuerung der Lokomotiven Einfluß genommen.

Wichtiger Bestandteil der Steuerung ist eine dynamische Datenbank, in der Informationen über den jeweiligen Status der Anlage enthalten sind. Die Datenbank wird während des Ablaufs durch Informationen über die Rückmeldemodule ständig aktualisiert und beeinflußt dadurch den Fahrbetrieb.
Zu dieser Datenbank gehören die Fakten: zug_kommt(Richtung), gleis_frei(Gleis), unfall.

Das Prädikat aktualisiere_gleisbelegung wertet die Informationen vom Rückmeldemodul aus und fügt einen Fakt gleis_frei(Gleis) ein bzw. löscht einen solchen Fakt, je nachdem ob

ein Zug ein oder ausgefahren ist. Gleis ist dabei an die Gleisnummer gebunden, auf die sich die Aktualisierung bezieht. Die Datenbank kann also zwei Fakten enthalten (gleis_frei(1); gleis_frei(2)) aber auch keinen Fakt, wenn beide Gleise belegt sind. Die Informationen über die Gleisbelegung werden mit Hilfe zweier Kontaktgleispaare ermittelt. Zur Feststellung ob ein Zug aus- oder eingefahren ist, dienen die Schaltgleise, die sich an der Nord- und der Südseite des Bahnhofs befinden. Das Rückmeldemodul registriert, wenn eine Lok über das Schaltgleis gefahren ist. Die Fahrtrichtung kann ebenfalls ermittelt werden, da unterschiedliche Richtungen zur Kontaktgabe verschiedener Kontakt führen. aktualisiere_strecke fügt zur Datenbasis einen Fakt zug_kommt(norden) hinzu, wenn ein Zug über Schaltgleis B in Südrichtung ausfährt. Dieser Fakt wird aus der Datenbank wieder entfernt, wenn der Zug über Schaltgleis A wieder in den Bahnhof einfährt. Das gleiche geschieht für die Aus- und Einfahrten in Nordrichtung. Der Fakt unfall wird dann zur Datenbasis hinzugefügt, wenn ein Zug in Nord- und ein zweiter in Südrichtung unterwegs ist. aktualisiere_gleisbelegung und aktualisiere_Strecke sind in das Prädikat bahnstatus integriert, dessen Aufruf zur Überarbeitung der gesamten Datenbasis führt.

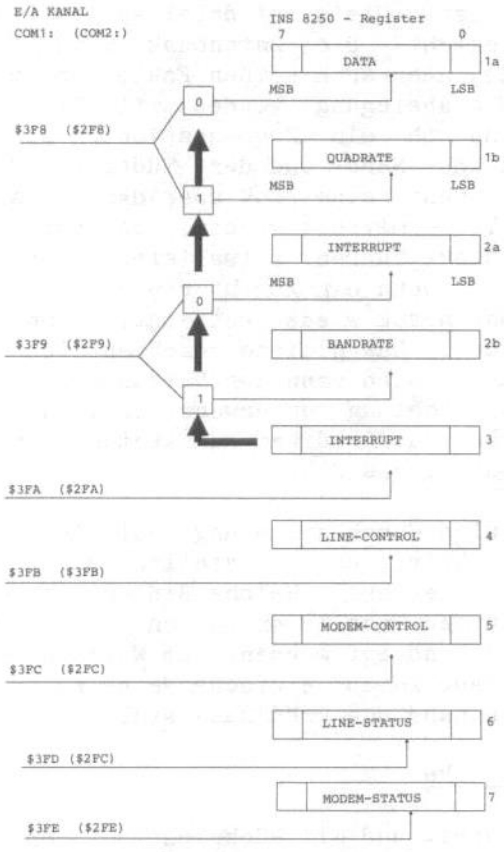
Die vollständige Steuerung wird durch steuerung realisiert, das neben bahnstatus ein zweites wichtiges Prädikat enthält, nämlich stellen. Durch stellen wird die Ansteuerung der Signale und Weichen durchgeführt. Welche Bedingungen zu einzelnen Weichen- bzw. Signalstellungen führen, kann der Tabelle entnommen werden. Dabei wird deutlich, daß nur Informationen der Datenbasis benötigt werden, um Weichen und Signale zu beeinflussen. Prolog ermöglicht es, die Tabelle auf einfache Weise zu programmieren (programming in logic). Der Leser möge sich anhand des Prädikats stellen selbst davon überzeugen.

Programmierhinweise und Ausblicke

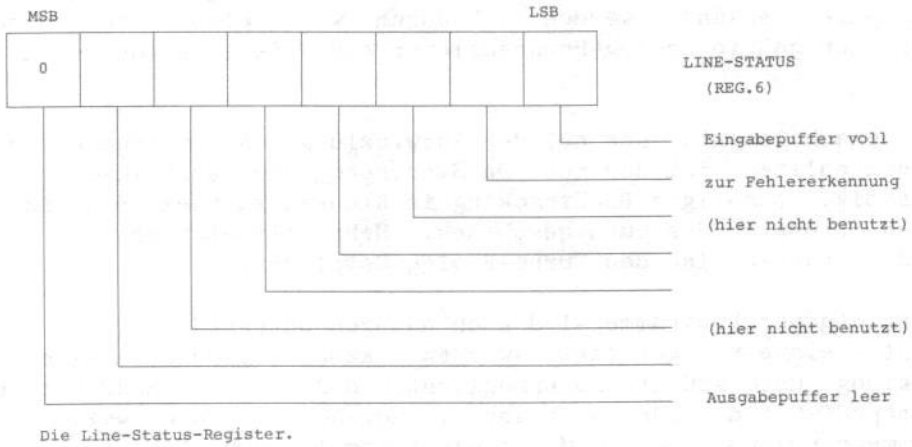
Aus Gründen der Übersichtlichkeit und mit Rücksicht auf den Prolog-Anfänger wurde in dem Programm darauf verzichtet, intensiv von Datentyp-Deklarationen Gebrauch zu machen. Das birgt den Nachteil, daß in den Prädikaten häufig der Standardtyp symbol benutzt wird. Das führt letztlich dazu, daß Tippfehler bei der Verwendung derartiger Prädikate nicht durch den Compiler erkannt werden. Dadurch wird eine vermeidbare Fehlerquelle geschaffen, die der geübte Prolog-Programmierer durch Deklaration geeigneter Datentypen umgehen kann.

Ein weiteres Sprachelement, das bei der Entwicklung von Programmen sehr sorgfältig eingesetzt werden sollte, ist der cut. Da Steuerprogramme auch immer zeitkritisch sind, ist es häufig nötig, unnötiges Backtracking in Klausen zu vermeiden. Das kann nur durch den geschickten Einsatz des cuts geschehen. Sehr hilfreich bei der Optimierung der Programme ist die trace-Option des Turbo-Prolog-Compilers.

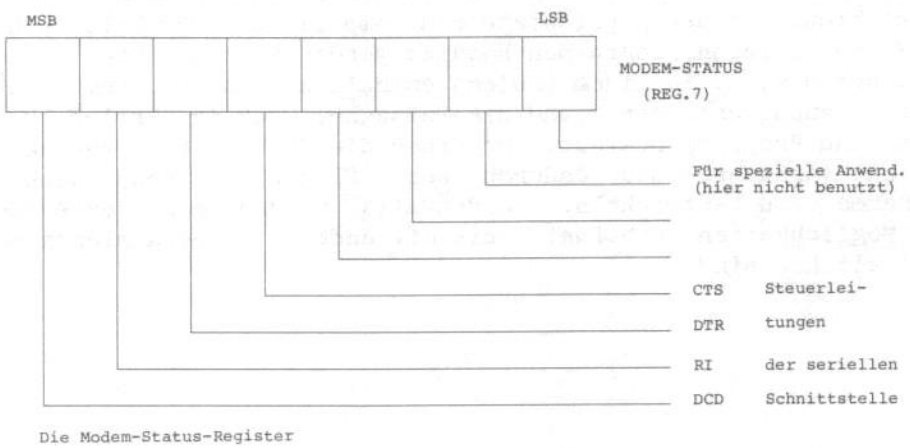
Die Entwicklung eigener Programme wird auch dadurch unterstützt, daß im Dialogbetrieb jedes Prädikat einzeln getestet werden kann, dadurch wird die modulare Programmentwicklung unterstützt und ermöglicht, daß einzelne Prädikate sorgfältig auf Korrektheit überprüft werden können, bevor sie auf höherer Ebene weiterverwendet werden. In diesem Zusammenhang muß auch die Möglichkeit des Turbo-Prolog-Systems erwähnt werden, Projects zu erstellen, in denen getestete und compilierte Prädikate enthalten sind, die auf einfache Weise in neuen Programmen benutzt werden können. Die Grundprädikate zur Bahnsteuerung könnten z. B. in einem Project enthalten sein. Die komfortabele Behandlung von Datenbanken (auch externer) und die Tatsache, daß in Prolog kein Unterschied zwischen Daten und Programm besteht, eröffnen die Möglichkeit Prädikate während des Programmlaufs zu verändern und dadurch sehr flexible, wenn nicht kybernetische Steuerungsprogramme zu entwickeln. Jedenfalls werden dem experimentierfreudigen Programmierer Möglichkeiten eröffnet, die mit anderen Programmiersprachen nur sehr schwer zu verwirklichen sind.



Zuordnung der Register des seriellen Portbausteins INS 8250 zu den Adressen der E/A-Kanäle des Computersystems.

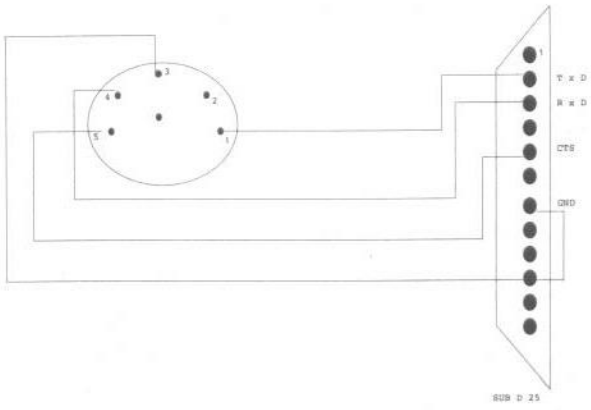


Die Line-Status-Register.

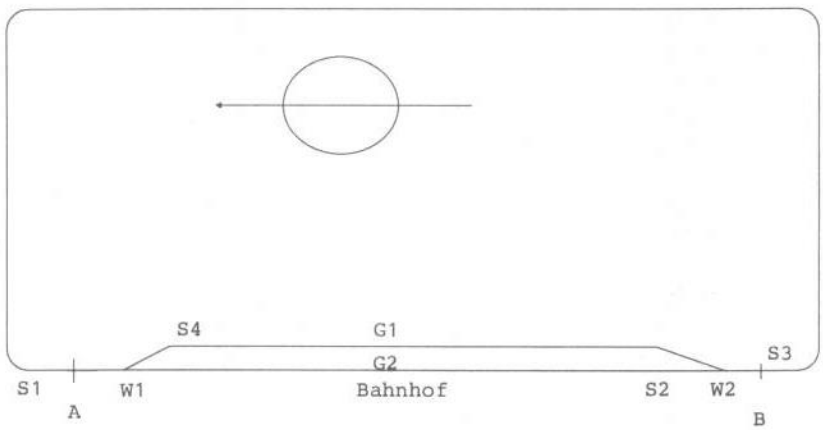


Die Modem-Status-Register

Die Bedeutung der Bits des LINE-Status- und MODEM-Status-Registers.



So müssen Digital-Interface und V 24 - Schnittstelle miteinander verbunden werden.



Das Gleissystem.

Die Tabelle gibt an, wie die einzelnen Register des Bausteins INS 8250 geladen werden müssen, um die Übertragungsparameter einzustellen.

Register	Wert
1b	$20
2b	0
2a	0
4	7

Die einzelnen Schalter des Rückmeldemoduls und deren Einfluß auf das Statuswort.

Bitnummer des Statusworts		Kontakt
MSB	15	
	14	
	13	Schaltgleis A nach Noren
	12	Schaltgleis A nach Süden
		Kontaktgleis G1
		Kontaktgleis G2
		Schaltgleis B nach Norden
		Schaltgleis B nach Süden
	7	
	.	
	.	
LSB	0	nicht benutzt

Die Tabelle zeigt, in welcher Weise Situationen auf der Gleisanlage die Signale und Weichen beeinflussen.

Zug kommt	weder Norden noch Süden		Norden		Süden		Norden und Süden
Gleis frei							
weder G1 noch G2	S1	rot	S1	rot	S1	rot	
	S2	grün	S2	grün	S2	rot	
	S3	rot	S3	rot	S3	rot	
	S4	grün	S4	rot	S4	grün	Stop
	W1	links	W1	rechts	W1	links	
	W2	links	W2	links	W2	rechts	
G1	S1	rot	S1	rot	S1	rot	
	S2	grün	S2	grün	S2	rot	
	S3	grün	S3	grün	S3	grün	
	S4	grün	S4	rot	S4	grün	Stop
	W1	links	W1	rechts	W1	links	
	W2	links	W2	links	W2	rechts	
G2	S1	grün	S1	grün	S1	grün	
	S2	grün	S2	grün	S2	rot	
	S3	rot	S3	rot	S3	rot	Stop
	S4	grün	S4	rot	S4	grün	
	W1	links	W1	rechts	W1	links	
	W2	rechts	W2	rechts	W2	rechts	
G1 und G2	S1	grün	S1	grün	S1	grün	
	S2	grün	S2	grün	S2	rot	
	S3	grün	S3	grün	S3	grün	
	S4	grün	S4	rot	S4	grün	Stop
	W1	rechts	W1	rechts	W1	rechts	
	W2	rechts	W2	rechts	W2	rechts	

```
        /* Ullrich Thiemann, In der Rehre 26, 3000 Hannover 91 */

        /* Demonstrationsprogramm zum Einsatz von Turbo Prolog */
        /* zur Steuerung einer Bahnanlage von MÄRKLIN Digital. */

domains
   werteliste = integer*

database
   gleis_frei(integer)
   zug_kommt(symbol)
   unfall

predicates
/*************************************************************************/
/*                                                                       */
/*              Prädikate zur Bedienung der Schnittstelle                */
/*                                                                       */
/*************************************************************************/


   warte_auf_eingabe
   warte_auf_ausgabe
   ausgabe_möglich
   cts
```

```
   epv
   spl
   s_read(integer)
   s_write(integer)

/*************************************************************************/
/*                                                                       */
/*            Grundprädikate zur Steuerung des Bahnbetriebes             */
/*                                                                       */
/*************************************************************************/

   init_v24
   warten(integer)
   noch_nicht(integer)
   zuordnung(symbol, integer)
   stop
   go
   fahrstufe(symbol, integer, symbol)
   fahrtrichtung(symbol)
   weiche_stellen(symbol, symbol)
   signal_stellen(symbol, symbol)
   hole_status(integer, werteliste)
   erstelle_liste(integer, werteliste)
   umwandeln(werteliste, werteliste)

/*************************************************************************/
/*                                                                       */
/*                    Prädikate für eine einfache Steuerung              */
/*                                                                       */
/*************************************************************************/

   repeat
   bahnstatus
   aktualisiere_gleisbelegung(integer)
   aktualisiere_strecke(integer)
   aktualisiere_norden(integer)
   aktualisiere_süden(integer)
   stellen
   steuerung
   ende

goal
        makewindow(1, 7, 0, "", 0, 0, 25, 80), clearwindow,
        makewindow(2, 7, 15, " PROLOG-Demonstration zu MÄRKLIN-Digital ",
                   0,0,5,80), cursor(2,21),
        write("Abbrechen durch Betätigen von BREAK!"),
        repeat, warten(1), steuerung, ende, !.

clauses
/*************************************************************************/
/*                                                                       */
/*                Prädikate zur Bedienung der Schnittstelle              */
/*                                                                       */
/*************************************************************************/

/* ----------------- ermittelt den Status der CTS-Leitung ---------------- */

   cts :-
        port_byte($3fe, Status), bitand(Status, $10, S), S = $10.

/* ------------ ist erfüllt, wenn ein Zeichen empfangen wurde ------------ */

   epv :-
```

```
        port_byte($3fd, Status), bitand(Status, 1, S), S = 1.

/* --------- ist erfüllt, wenn ein Zeichen gesendet werten kann ---------- */

   spl :-
        port_byte($3fd, Status), bitand(Status, $40, S), S = $40.

/* ---------------- wartet bis ein Zeichen empfangen wurde -------------- */

   warte_auf_eingabe :- !, not(epv), warte_auf_eingabe.

/* ------------ wartet bis ein Zeichen ausgegeben werden kann ----------- */

   warte_auf_ausgabe :- !, not(ausgabe_möglich), warte_auf_ausgabe.

   ausgabe_möglich :- cts, spl.

/* ---------------gibt ein Zeichen an die Schnittstelle aus.------------- */

   s_write(Wert) :-
        not(warte_auf_ausgabe), port_byte($3f8, Wert).

/* --------------liest ein Zeichen von der Schnittstelle ein.------------ */

   s_read(Wert) :-
        not(warte_auf_eingabe), port_byte($3f8, Wert).

/*************************************************************************/
/*                                                                       */
/*            Grundprädikate zur Steuerung des Bahnbetriebes             */
/*                                                                       */
/*************************************************************************/

/* ------------ zur Realisierung von Wartezeiten in Sekunden ----------- */

   warten(0).
   warten(Zeit) :-
        time(_,_,Sek,_), Neue_Zeit = (Zeit + Sek) mod 60,
        not(noch_nicht(Neue_Zeit)).

   noch_nicht(Neue_Zeit) :- !,
        time(_,_,Sek,_), Sek < Neue_Zeit, noch_nicht(Neue_Zeit).

/* -------- initialisiert Com2:, Übertragungsprotokoll: 2400,N,8,2 ------- */

   init_v24 :-
        port_byte($3fb, $80), port_byte($3f8, $30), port_byte($3f9, 0),
        port_byte($3fb, 7), port_byte($3f9, 0), port_byte($3f8, _).

/*  die folgenden Fakten legen Adressen der verschiedenen  Einheiten fest  */

   zuordnung(l1, 26).
   zuordnung(l2, 27).
   zuordnung(w1, 7).
   zuordnung(w2, 8).
   zuordnung(s1, 1).
   zuordnung(s2, 2).
   zuordnung(s3, 3).
   zuordnung(s4, 4).

/* ------------ stellt die Fahrstufe der ausgewählten Lok ein ----------- */

   fahrstufe(Nr, Stufe, Funktion) :-
```

```
        Funktion = aus, Stufe >= 0, Stufe < 15,
        zuordnung(Nr, Lokadresse), s_write(Stufe), s_write(Lokadresse).
   fahrstufe(Nr, Stufe, Funktion) :-
        Funktion = ein, Stufe >= 0, Stufe < 15, Ausgabe = Stufe + 16,
        zuordnung(Nr, Lokadresse), s_write(Ausgabe), s_write(Lokadresse).

/* ---------- schaltet die Fahrtrichtung der ausgewählten Lok um --------- */

   fahrtrichtung(Nr) :-
        zuordnung(Nr, Lokadresse), s_write(15), s_write(Lokadresse).

/* ------------------------- stellt die Weichen -------------------------- */

   weiche_stellen(Nr, Stellung) :-
        Stellung = links, zuordnung(Nr, Adresse), s_write(33),
        s_write(Adresse).
   weiche_stellen(Nr, Stellung) :-
        Stellung = rechts, zuordnung(Nr, Adresse), s_write(34),
        s_write(Adresse).

/* ------------------------- stellt die Signale -------------------------- */

   signal_stellen(Nr, Stellung) :-
        Stellung = grün, zuordnung(Nr, Adresse), s_write(33),
        s_write(Adresse).
   signal_stellen(Nr, Stellung) :-
        Stellung = rot, zuordnung(Nr, Adresse), s_write(34),
        s_write(Adresse).

/* ------------------------------- Notaus ------------------------------- */

   stop :-  not(warte_auf_ausgabe), port_byte($3f8, 97).

/* --------------------- Freigabe der Anlage nach Notaus ---------------- */

   go :- not(warte_auf_ausgabe), port_byte($3f8, 96).

/* ---------------------- fragt Rückmeldemodule ab --------------------- */

   hole_status(Modul, Werteliste) :-
        Modul > 0, Modul < 32, Moduladresse = 128 + Modul, Länge = 2 * Modul,
        port_byte($3f8, _), s_write(Moduladresse),
        erstelle_liste(Länge, Hilfsliste), umwandeln(Hilfsliste, Werteliste).

   erstelle_liste(0,[]) :- !.
   erstelle_liste(Länge, Werteliste) :-
        s_read(Wert), N_Länge = Länge - 1, erstelle_liste(N_Länge, Teilliste),
        Werteliste = [Wert|Teilliste].

   umwandeln([], []).
   umwandeln(Hilfsliste, Werteliste) :-
        Hilfsliste = [Lo_Byte, Hi_Byte | Tail], umwandeln(Tail, Teilliste),
        Wert = Lo_Byte + $100 * Hi_Byte, Werteliste = [Wert|Teilliste].

/*************************************************************************/
/*                                                                       */
/*                        Beispiel für eine Steuerung                    */
/*                                                                       */
/*************************************************************************/

/* ----------------------- ein Hilfsprädikat --------------------------- */

   repeat.
```

```
   repeat :- repeat.


/* ------ ermittelt den Bahnstatus und aktualisiert die Datenbank -------- */

   bahnstatus :-
        hole_status(1, Statusliste), Statusliste = [Wort|_],
        bitand(Wort, $C00, G), aktualisiere_gleisbelegung(G),
        bitand(Wort, $3300, S), aktualisiere_strecke(S).

   aktualisiere_gleisbelegung(G) :-
        G = 0, not(gleis_frei(1)), asserta(gleis_frei(1)), fail;
        G = 0, not(gleis_frei(2)), assertz(gleis_frei(2)), !;
        G = $800, retract(gleis_frei(1)), fail;
        G = $800, not(gleis_frei(2)), asserta(gleis_frei(2)), !;
        G = $400, retract(gleis_frei(2)), fail;
        G = $400, not(gleis_frei(1)), assertz(gleis_frei(1)), !;
        G = $C00, retract(gleis_frei(1)), fail;
        G = $C00, retract(gleis_frei(2)).
   aktualisiere_gleisbelegung(_).

   aktualisiere_strecke(S) :-
        bitand(S, $3000, A_S1), aktualisiere_norden(A_S1),
        bitand(S, $300, A_S2), aktualisiere_süden(A_S2).

   aktualisiere_norden(S) :-
        S = $1000, !, retract(zug_kommt(norden));
        S = $2000, !, asserta(zug_kommt(süden));
        S = $3000, !, asserta(zug_kommt(süden)), retract(zug_kommt(norden)).
   aktualisiere_norden(_).

   aktualisiere_süden(S) :-
        S = $100, !, assertz(zug_kommt(norden));
        S = $200, !, retract(zug_kommt(süden));
        S = $300, !, assertz(zug_kommt(norden)), retract(zug_kommt(süden)).
   aktualisiere_süden(_).

/* ------------------------- stellen der Signale ----------------------- */

   stellen :-
        gleis_frei(2), signal_stellen(s1, grün), fail;
        not(gleis_frei(2)), signal_stellen(s1, rot), fail;
        zug_kommt(süden), signal_stellen(s2, rot),
        weiche_stellen(w2, rechts), fail;
        not(zug_kommt(süden)), signal_stellen(s2, grün), fail;
        gleis_frei(1), signal_stellen(s3, grün), fail;
        not(gleis_frei(1)), signal_stellen(s3, rot), fail;
        zug_kommt(norden), signal_stellen(s4, rot),
        weiche_stellen(w1, rechts), fail;
        not(zug_kommt(norden)), signal_stellen(s4, grün), fail;
        not(zug_kommt(norden)), not(gleis_frei(1)),
        weiche_stellen(w1, links), fail;
        not(zug_kommt(süden)), not(gleis_frei(2)),
        weiche_stellen(w2, links), fail;
        zug_kommt(norden), zug_kommt(süden), stop, asserta(unfall).


/* ------------------ übernimmt die Steuerung des Systems ---------------- */

   steuerung :-
        init_v24, repeat, bahnstatus, stellen, ende, !.

   ende :- unfall.
```

Fahrwegoptimierung

Das Programm, wie es in der jetzigen Version vorliegt, ist für den HP 200 und den ATARI ST geschrieben. Außerdem gibt es auch eine angepaßte Version für den ATARI ST.
Das Programm hat folgende Fähigkeiten:

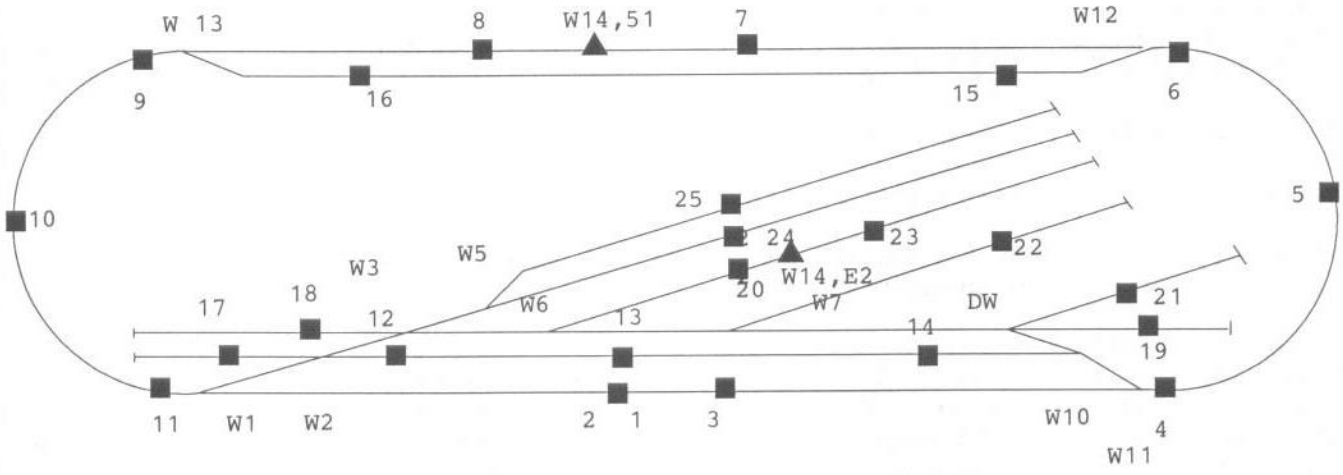
- Eingabe einer beliebigen Kursstrecke über mehrere Kontakte

- Suche des kürzesten Weges zwischen den eingegebenen Kontakten

- Graphische Simulation der Fahrt auf dem Bildschirm

- Tatsächliche Fahrt auf der Anlage und gleichzeitiges Darstellen der Fahrt auf dem Bildschirm

Das zentrale Problem, welches dieses Programm zu bewältigen hat, besteht darin, den kürzesten Weg zwischen zwei Kontakten zu finden und ihn in eine Form zu bringen, die dem Computer die Simulation am Bildschirm, sowie die Fahrt der Lok auf der Anlage ermöglicht.



Realisiert wurde folgende Möglichkeit:

Herausfinden des kürzesten Weges durch rekursive Programmierung. Um dies näher zu erläutern, muß ich jedoch zuerst den Grundgedanken, der diesem Algorithmus zugrunde liegt, genauer beschreiben:

Wenn man zu jedem Kontakt die nächstgelegenen Kontakte mitsamt Fahrtrichtung und notwendigen Weichenstellungen angibt ist es möglich, durch eine Aneinanderreihung dieser "Primärkursstrecken" jeden beliebigen Kontakt anzufahren.

Um dies zu realisieren, wird ein Stringfeld mit folgendem Aufbau geschaffen:

- Für jeden Kontakt existiert ein Feldelement C$ (Kont.Nr.), bei dem angegebenen Beispielkurs also insgesamt 25 Elemente.

1. Beispiel:
Für Kontakt Nr.3 hat dieses Element folgendes Aussehen:

```
C$(3)="2,   11,2,0,   4,1,0"
```

Die erste Zahl gibt hierbei die Anzahl der als nächstes erreichbaren Kontakte an. es sind also zwei Kontakte erreichbar.

Die zweite Zahl ist die Kontaktnummer des ersten erreichbaren Kontaktes, also Nr.11. Die nächste Zahl gibt die Fahrtrichtung der Lok an, die notwendig ist, um diesen Kontakt zu erreichen. Die Null schließlich zeigt an, daß keine Weichenstellungen notwendig sind, um diesen Kontakt zu erreichen.

Die nächste Zahl gibt den zweiten erreichbaren Kontakt an, die Nr. 4. Die Fahrtrichtung ist dabei genau umgekehrt wie vorher. Die Null zeigt wiederum an, daß keine Weichen zu stellen sind.

2.Beispiel:
Für Kontakt Nr.4 sieht das Stringelement folgendermaßen aus:

```
c$(4)="5,  3,2,1,11,G,  14,2,2,11,R,10,R,  18,2,3,11,R,10,G,3,G,..."
```

die erste Zahl gibt wieder die Anzahl der erreichbaren Kontakte an, in diesem Beispiel also fünf. Erster erreichbarer Kontakt ist die Nummer drei, Fahrtrichtung, um diesen Kontakt zu erreichen ist zwei, dazu ist eine Weichenstellung nötig, nämlich Weiche 11 auf (G)erade.
Zweiter erreichbarer Kontakt ist Nr. 14, Fahrtrichtung dazu ist zwei, zwei Weichenstellungen sind nötig: Weiche 11 (R)und, Weiche 10 (R)und.
Dritter Kontakt ist Nr. 18, Fahrtrichtung 2, 3 Weichenstellungen, nämlich Nr. 11 (R)und, Nr. 10 (G)erade und Nr. 3 (G)erade.

Nach diesem Muster sind nun sämtliche Primärkursstrecken aufgebaut.

Nun zum Suchalgorithmus. Hierzu ebenfalls ein Beispiel:

Nehmen wir an, ich will die Lok von Kontakt Nr. 10 nach Kontakt Nr. 15 fahren lassen, so geht der Algorithmus folgendermaßen vor:

Erklärung zu Teil A:
Ausgehend von Kontakt Nr. 10 liegt als erster erreichbarer Kontakt Nr. 11 vor. Von diesem aus sind dann Nr. 3, Nr. 12, Nr. 25 usw. erreichbar. davon ausgehend kann man von jedem einzelnen wieder eine gewisse Anzahl von Kontakten erreichen. Es ist klar, daß man, führt man dieses Schema weiter nach einer endliche Anzahl von schritten auf den Zielkontakt Nr. 15 kommt.

Der zweite mögliche Kontakt, der von Nr. 10 aus zu erreichen ist, ist die Nr. 9.

Erklärung zu Teil B:
Von Nr. 9 ausgehend kann nun Nr.8, dann über Nr. 6 schließlich Kontakt Nr. 15 erreicht werden.

Erklärung zu Teil C:
Von Nr. 9 kann jedoch auch Nr. 16 und schließlich Nr. 15 erreicht werden. da dies offensichtlich kürzeste Weg ist, wird dieser schließlich auch gefahren.

Die einzelnen Verbindungsstrecken stellen eine sog. Baumstruktur dar, die der Algorithmus durchgeht. Hat er eine Strecke gefunden, die auf den Zielkontakt führt, so wird diese in einem sogenannten "Momentankursstring" zwischengespeichert und der Algorithmus sucht weiter nach einem möglicherweise kürzeren Wegg.

Dabei sucht er den Baum aber nicht komplett ab, sondern geht nur noch so tief, wie es nötig ist. Hat er also einmal eine Strecke gefunden, bei der die Lok z.B. 3 Kontakte überfahren muß, sucht der algorithmus den Baum nur noch bis zu einer Maximalsuchtiefe 2 ab, da bei einer größeren suchtiefe sowieso nur längere strecken gefunden werden können, die zum Ziel führen.

Das kürzeste gefundene Fahrfeld wird dann in einem sog. "Fahrstring" umgewandelt. Dieser hat folgendes Format:
Bsp.:

Kurs von Kont. Nr. 10 nach Kont.Nr. 15
F$= "9, 2, 0, 16, 2, 1, 13,R 15,2,0"

Die erste Zahl gibt den nächsten Kontakt an, der gefahren wird, also die Nr. 9. Die zweite Zahl gibt die Fahrtrichtung an, also zwei. Um Kontakt Nr. 9 zu erreichen sind keine Weichenstellungen nötig (dritte Zahl).
Der zweite Kontakt, der angefahren werden muß, ist Nr. 16. Fahrtrichtung ist zwei, eine Weichenstellung ist nötig und zwar Weiche Nr. 13 (R)und.
Anschließend kann Nr 15 erreicht werden. Fahrtrichtung 2. Keine Weichenstellung notwendig.

Hat man nun diesen Fahrstring, so ist eine Simulation am Bildschirm und auch die Fahrt auf der Anlage kein Problem mehr. Es muß lediglich eine softwaremäßige Schnittstelle zu Märklin-Digital geschaffen werden, die folgende Kommandos "versteht":

- Lok beschleunigen, abbremsen und Richtungsumkehr
- Weiche schalten
- Warten, bis bestimmte Kontaktnummer erreicht ist.

Nachfolgend ein Auszug des Programms. Das gesamte Programm-Listing erhalten Sie gegen einen frankierten und adressierten Umschlag von der Redaktion.

```
3730  !
3740  SUB Suchkurs(INTEGER Ausg_kont,INTEGER Ziel_kont)
3741    COM /Liste1/ A$[50],B$[50],F$[750],C$(1:25)[750]
3742    COM /Liste2/ INTEGER F_feld_p1,INTEGER F_feld1(25)
3752    COM /Liste3/ INTEGER F_feld_p,INTEGER F_feld(25),INTEGER Richt,
                     INTEGER Anz_weich,INTEGER F,INTEGER Z,Kont$[250]
3753    INTEGER Tiefe,Un_grenze,Kont_nr
3773  !
3774    PRINT CHR$(12)
3776    FOR I=0 TO 15
3777      PRINT
3778    NEXT I
3780    PRINT "SUCHE KURS VON KONTAKT NR. ";Ausg_kont;" NACH KONTAKT NR. ";Ziel_kont
3791    PRINT
3810    F_feld(0)=Ausg_kont
3820    F_feld_p=1
3830    F_feld_p1=25
3840    CALL Follow_kont(Ausg_kont,(Tiefe),Ziel_kont)
3860  ! PRINT "FAHRSTRECKE:"
3870  ! FOR Z=0 TO F_feld_p1-1
3880  !   PRINT "NUMMER,KONTAKT: ";Z,F_feld1(Z)
3890  ! NEXT Z
3900  ! PRINT
3910  SUBEND
3920  !
3921  !
3922  !
3930  SUB Follow_kont(INTEGER Ur_kont,INTEGER Tiefe,INTEGER Ziel_kont)
3931    COM /Liste1/ A$[50],B$[50],F$[750],C$(1:25)[750]
3932    COM /Liste2/ INTEGER F_feld_p1,INTEGER F_feld1(25)
3933    COM /Liste3/ INTEGER F_feld_p,INTEGER F_feld(25),INTEGER Richt,
                     INTEGER Anz_weich,INTEGER F,INTEGER Z,Kont$[250]
3934    INTEGER Kont_pointer,Anz_kont,Un_grenze,Kont_nr
3936    !
3940    !REKURSIVE PROCEDUR ZUR ERMITTLUNG DER KURSSTRECKE
3941    !
3947    !
```

```
3960    Tiefe=Tiefe+1
3970    !PRINT
3980    !PRINT "AUGENBLICKLICHE TIEFE: ";Tiefe
3990    !
3991    Un_grenze=0
4000    CALL Dat(C$(Ur_kont),Un_grenze,B$)
4010    Anz_kont=VAL(B$)
4020    Kont_pointer=Un_grenze
4030    !
4040    IF Tiefe<F_feld_p1-1 AND Tiefe<=6 THEN
4080    !
4090      FOR I=1 TO Anz_kont
4100        REPEAT
4110          F=0
4120          CALL Lese_kontstring(Kont_pointer,Ur_kont,Kont$,Kont_nr,Richt,Anz_weich)
4130          IF Kont$="" THEN
4131            F=0
4141          ELSE
4150            CALL Pruef_kontakt(Kont_nr)
4160           !PRINT "KONTAKTNR. MUß WEITERVERFOLG WERDEN?(1=NEIN): ";Kont_nr,F
4161          END IF
4162        !
4170        UNTIL F=0
4180        !
4190        IF Kont$<>"" THEN
4230          CALL Dat(Kont$,0,B$)
4240          CALL Kont_nr_feld(Kont_nr)
4250          !
4260          !PRINT "FAHRFELD:"
4270          !FOR Z=0 TO F_feld_p-1
4280          !PRINT F_feld(Z);
4290          !NEXT Z
4300          !PRINT
4320          !
4330          IF Kont_nr=Ziel_kont THEN
4340            CALL Kont_vergleich
4350          END IF
4360          !
4370          CALL Dat(C$(Kont_nr),0,B$)
4380          CALL Follow_kont(Kont_nr,(Tiefe),Ziel_kont)
4390          !
4391        END IF
4400      !
4410      NEXT I
4420    !
4421    END IF
4440    F_feld(F_feld_p-1)=0
4450    F_feld_p=F_feld_p-1
4460    !
4470    ! PRINT "RUECKSPRUNG AUS FOLLOW KONT"
4480  SUBEND
4490    !
4500    !
4510  SUB Kont_vergleich
4520    !VERGLEICHT FAHRSTRECKE MIT BISHERIGER KÜRZESTER FAHRSTRECKE
4521    COM /Liste2/ INTEGER F_feld_p1,INTEGER F_feld1(25)
4522    COM /Liste3/ INTEGER F_feld_p,INTEGER F_feld(25),INTEGER Richt,
                     INTEGER Anz_weich,INTEGER F,INTEGER Z,Kont$[250]
4532    INTEGER I
4542    !
4552    !
4562    ! PRINT "STRECKENLÄNGENVERGLEICH!"
4572    !
```

```
4582    IF F_feld_p1>F_feld_p THEN
4592      FOR I=0 TO F_feld_p1
4602        F_feld1(I)=F_feld(I)
4612      NEXT I
4622      F_feld_p1=F_feld_p
4632      !
4642      !
4652      !  PRINT "NEUE KÜRZESTE FAHRSTRECKE GEFUNDEN!!"
4662      !   FOR Z=0 TO F_feld_p1-1
4672      !   PRINT "NUMMER,KONTAKT:",Z,F_feld1(Z)
4682      !  NEXT Z
4692      !  PRINT
4702      ! INPUT Dummy
4712    END IF
4732  SUBEND
4742  !
4752  !
4762  SUB Pruef_kontakt(INTEGER Nr)
4772    COM /Liste3/ INTEGER F_feld_p,INTEGER F_feld(25),INTEGER Richt,
                    INTEGER Anz_weich,INTEGER F,INTEGER Z,Kont$[250]
4782    !
4792    !PRUEFT OB KONTAKT SCHON EXISTIERT
4802    INTEGER I
4812    !
4822    FOR I=0 TO F_feld_p-1
4832      IF F_feld(I)=Nr THEN
4842        F=1
4852      END IF
4862    NEXT I
4872  !
4882  SUBEND
4892  !
4902  !
4912  SUB Kont_nr_feld(INTEGER Nr)
4913    COM /Liste3/ INTEGER F_feld_p,INTEGER F_feld(25),INTEGER Richt,
                    INTEGER Anz_weich,INTEGER F,INTEGER Z,Kont$[250]
4923    INTEGER I
4933    !
4943    F_feld(F_feld_p)=Nr
4953    F_feld_p=F_feld_p+1
4963  SUBEND
4973  !
4983  !
4993  SUB Make_fahrstring
5000    COM /Liste2/ INTEGER F_feld_p1,INTEGER F_feld1(25)
5010    COM /Liste1/ A$[50],B$[50],F$[750],C$(1:25)[750]
5015    COM /Liste3/ INTEGER F_feld_p,INTEGER F_feld(25),INTEGER Richt,
                    INTEGER Anz_weich,INTEGER F,INTEGER Z,Kont$[250]
5020    INTEGER I,Kont_pointer,Un_grenze,Ur_kont,Kont_nr
5030    !
5040    ! PRODUZIERT FAHRSTRING
5050    !
5060    FOR I=0 TO F_feld_p1-1
5061      Un_grenze=0
5070      CALL Dat(C$(F_feld1(I)),Un_grenze,B$)
5090      WHILE Kont_nr<>F_feld1(I+1)
5100        Ur_kont=F_feld1(I)
5110        CALL Lese_kontstring(Un_grenze,Ur_kont,Kont$,Kont_nr,Richt,Anz_weich)
5120      END WHILE
5130      F$=F$&Kont$
5140    NEXT I
5150  SUBEND
5160  !
```

ESPV3 steuert die Modellbahn

Das Programm ESPV3 ist ein Eisenbahn-Steuerungs-Programm, mit ihm können Abläufe (Fahrpläne) vollautomatisch gesteuert werden. Schaltkontakte werden abgefragt, ausgewertet und entsprechende Ausgaben (Lok und Magnetartikel-Befehle) ausgegeben. Bis zu 5 Loks kann man gleichzeitig und unabhängig mit kontinuierlicher Beschleunigung oder Verzögerung (Massensimulierung) betreiben. Durch das Entfallen der Trennstellen im Signalbereich ist auch hier eine konstante Zugbeleuchtung vorhanden. Das komplette Programm teilt sich in 4 Hauptbereiche auf:

1. Unterprogramme
2. Initialisierung
3. Anlagenspezifische Fahrstraßen
4. Fahrplan

```
1000 '      ***   R O M A - S O F T   ***
1010 '      NAME: ESPV3        NR.: 860624
1020 '      COPYRIGHT C 1986 BY ROMA-SOFT
1030 '      LETZTE ÄNDERUNG:        861208
1040 '
1050 '
1060 '
1070 'EISENBAHN-STEUER-PROGRAMM, AUFGEBAUT IN UNTERPROGRAMM-TECHNIK
1080 '
1090 GOTO 2000 'INITIALISIERUNG
1100 '------------------------- SEND2
1110 OUT &H29,&H10:WAIT &H29,&H20:OUT &H29,&H10
1120 OUT &H28,DB:OUT &H28,AB
1130 RETURN
1140 '------------------------- SEND1
1150 OUT &H29,&H10:WAIT &H29,&H20:OUT &H29,&H10
1160 OUT &H28,SB
1170 RETURN
1180 '------------------------- LOKBE
1190 FOR I=1 TO NL
1200 IF L(I,VS)=L(I,VI) THEN 1250
1210 IF (L(I,VS) AND 16)<>(L(I,VI) AND 16) THEN L(I,VI)=L(I,VI) XOR 16
1220 IF L(I,VS)-L(I,VI)>L(I,MB) THEN L(I,VI)=L(I,VI)+L(I,MB):GOTO 1250
1230 IF L(I,VS)-L(I,VI)<-L(I,MV) THEN L(I,VI)=L(I,VI)-L(I,MV):GOTO 1250
1240 L(I,VI)=L(I,VS)
1250 DB=INT(L(I,VI)):AB=L(I,NR)
1260 GOSUB 1110 'SEND2
1270 NEXT I
1280 RETURN
1290 '------------------------- RMLE
1300 GOSUB 1190 'LOKBE
1310 SB=RN:GOSUB 1150 'SEND1
1320 WAIT &H29,1
1330 FOR K=0 TO NB
1340 IF INP (&H29) AND 1=0 THEN 1340
1350 B(K)=INP (&H28)
1360 NEXT K
1370 RETURN
1380 '------------------------- SENDF
1390 AM=FS(SN,0)
1400 FOR I=1 TO AM
1410 DB=FS(SN,I*2):AB=FS(SN,I*2-1)
1420 GOSUB 1110 'SEND2
1430 GOSUB 1600 'ZEIT1
1440 NEXT I
1450 SB=L:GOSUB 1150 'SEND1
1460 RETURN
1470 '------------------------- LOKSS
```

```
1480 IF L(SS,VI)>16 THEN L(SS,VI)=16:L(SS,VS)=16 ELSE L(SS,VI)=0:L(SS,VS)=0
1490 GOSUB 1190 'LOKBE
1500 RETURN
1510 '------------------------- KONTA
1520 KN=KN-1
1530 IF FNK(KN)=0 THEN GOSUB 1300:GOTO 1530
1540 RETURN
1550 '------------------------- SENDM
1560 DB=S:AB=M:GOSUB 1110:GOSUB 1600 'SEND2, ZEIT1
1570 SB=L:GOSUB 1150 'SEND1
1580 RETURN
1590 '------------------------- ZEIT1
1600 FOR K=1 TO 50:NEXT K:RETURN
1610 '------------------------- ZBFSA
1620 L(LN,VI)=F:L(LN,VS)=F
1630 FOR K=1 TO Z:GOSUB 1190:NEXT K:RETURN
1640 '------------------------- FARWE
1650 IF L(FW,VI)>16 THEN L(FW,VI)=31:L(FW,VS)=31 ELSE L(FW,VI)=15:L(FW,VS)=15
1660 GOSUB 1190 'LOKBE
1670 IF L(FW,VI)>16 THEN L(FW,VI)=16:L(FW,VS)=16 ELSE L(FW,VI)=0:L(FW,VS)=0
1680 GOSUB 1190 'LOKBE
1690 RETURN
1700 '
2000 '------------------------- INITIALISIERUNG
2010 PRINT CHR$(12) 'CLEAR SCREEN
2020 FOR I=0 TO 2:PB=INP (&H28):NEXT I 'EMPFANGSPUFFER AUSRÄUMEN
2030 DEFINT A-K,M-Z
2040 '------------------------- INIT-LOKBE
2050 NL=6           'ANZAHL DER LOKS
2060 NF=5           'ANZAHL DER LOK-WERTE
2070 NR=1           'FELD 1=LOK-NR
2080 VS=2           'FELD 2=SOLL-GESCHW (FAHRPLAN)
2090 VI=3           'FELD 3=IST-GESCHW (SPEICHER)
2100 MB=4           'FELD 4=MSK-BESCHLEUNIGUNG
2110 MV=5           'FELD 5=MSK-VERZÜGERUNG
2120 DIM L(NL,NF)   'LOK-FELD
2130 L(1,NR)=20:L(1,MB)=.3:L(1,MV)=.7   'V 220
2140 L(2,NR)=11:L(2,MB)=.4:L(2,MV)=2    'VT 11.5
2150 L(3,NR)=3:L(3,MB)=.5:L(3,MV)=.2    'BR 03
2160 L(4,NR)=50:L(4,MB)=.5:L(4,MV)=.15  'BR 050
2170 L(5,NR)=73:L(5,MB)=.5:L(5,MV)=.5   'BR 86-1/2 U. UNI ADR
2180 L(6,NR)=80:L(6,MB)=.5:L(6,MV)=.5   'L(6,NR)=PSEUDO-LOK
2190 '------------------------- INIT-RMLE
2200 DIM M%(7):FOR I=0 TO 7:M(I)=2^(7-I):NEXT I 'FELD MIT WERTIGKEIT
2210 DIM B%(3)      '4 BYTE RESERVIERUNG FÜR RM
2220 DEF FNK(KN)=SGN(B(KN\8) AND M(KN MOD 8))   'SCHALTERAUSWERTUNG
2230 RN=130         '2 RM LESEN
2240 NB=3           'ANZAHL DER RM-BYTES (0-3 -->4 BYTES)
2250 RA=128         'RESET-MODE WIRD ABGESCHALTET
2260 RE=192         'RESET-MODE WIRD EINGESCHALTET
2270 '------------------------- INIT-MAG
2280 R=34           'MAG-ROT (RUND)
2290 G=33           'MAG-GRÜN (GERADE)
2300 L=32           'MAG-LÜSCHEN (SPULE ABSCHALTEN)
2310 '------------------------- INIT-SENDF
2320 NS=30          'ANZAHL DER FAHRSTRASSEN
2330 NM=20          '10 MAG PRO FS
2340 DIM FS(NS,NM) 'FS-FELD
2350 READ SN        'FS-NR
2360 IF SN=0 THEN 3000 'FAHRPLAN
2370 READ AM        'ANZAHL DER MAG PRO FS
2380 FS(SN,0)=AM
2390 FOR I=1 TO AM
```

Steuerungsprogramm

```
2400 READ A$
2410 MN=VAL(A$)     'MAG-NR AUS DATA-FELD
2420 IF RIGHT$(A$,1)="R" THEN MS=R ELSE MS=G
2430 FS(SN,I*2-1)=MN:FS(SN,I*2)=MS
2440 NEXT I
2450 GOTO 2350      'LESE NÄCHSTES DATA-FELD
2460 '------------------------- ANLAGEN-SPEZIFISCHE-FAHRSTRASSEN
2470 'FS-BEISPIEL:
2480 'DATA SN,AM,10G,11R     SN=FS-NR  AM=ANZAHL DER MAG PRO FS
2490 '                       10G=MAG-NR 10 MIT STELLUNG GRÜN
2500 '                       11R=MAG-NR 11 MIT STELLUNG ROT
2510 'DATA 0                 WIRD AM ENDE DER LETZTEN FS BENÜTIGT
2520 '
2530 DATA 01,01,11G
2540 DATA 02,02,11R,17G
2550 DATA 03,04,11R,17R,18G,19R
2560 DATA 04,04,11R,17R,18G,19G
2570 DATA 05,05,11R,17R,18R,20R,21R
2580 DATA 06,05,11R,17R,18R,20G,22R
2590 DATA 11,03,6R,30R,29G
2600 DATA 12,05,6R,30G,29G,28R,27G
2610 DATA 13,07,6R,30G,29G,28G,27G,26R,25G
2620 DATA 14,08,6R,30G,29G,28G,27G,26G,25G,23R
2630 DATA 15,02,30R,29R
2640 DATA 16,04,30G,29R,28R,27G
2650 DATA 17,06,30G,29R,28G,27G,26R,25G
2660 DATA 18,07,30G,29R,28G,27G,26G,25G,23R
2670 DATA 21,02,9G,10G
2680 DATA 22,02,9R,10R
2690 DATA 23,02,7G,8G
2700 DATA 24,02,7R,8R
2710 DATA 25,06,1G,2G,3G,4G,5G,6G
2720 DATA 26,05,1G,2G,3G,4G,5R
2730 DATA 27,04,1R,2R,3R,4R
2740 DATA 28,04,1G,2R,3G,4R
2750 DATA 0
3000 '------------------------- FAHRPLAN
```

1. Unterprogramme

SEND2 -- Sende 2 Byte zum Interface
In Zeile 1110 wird die CTS-Leitung (Freigabe-Leitung) des Interface abgefragt, d.h. wenn das Interface empfangsbereit ist, werden die 2-Bytes (DB=Datenbyte, AB=Adreßbyte) in Zeile 1120 ausgesendet.

SEND1 -- Sende 1 Byte zum Interface
In Zeile 1160 wird 1 Byte ausgesendet, sofern das Interface empfangsbereit ist (Zeile 1150).

LOKBE -- Lokberechnung
Berechnung für Anfahr- und Bremsverzögerung unter Berücksichtigung lokspezifischer MSK-Werten (MSK= Massensimulationskonstante). In dieser FOR-NEXT-Schleife werden zyklisch SOLL- und IST-Werte aller Loks verglichen, entsprechend den MSK-Werten angeglichen und zum Interface gesendet.
In dieses Unterprogramm wird vor jeder Kontaktabfrage verzweigt (siehe KONTA und RMLE). Die Reaktionszeitzeit bei der Kontaktabfrage ist stark abhängig von der Anzahl der angesteuerten Lokomotiven (je Lok ca. 55 ms). Bei 6 Loks ergibt das eine Zykluszeit (Reaktionszeit) f. Kontaktabfrage und Lokberechnung von ca. 330 ms. Dieses muß bei der Fahrplangestaltung berücksichsichtigt werden.

RMLE -- Rückmeldemodule lesen
Zum Senden des RM-Befehls in Zeile 1310 wir das Unterprogramm SEND1 verwendet. Mit dem WAIT-Befehl (1320) wird der Empfangspuffer im Z80-SIO Baustein abgefragt und solange

gewartet bis ein Byte empfangen wurde. die FOR-NEXT-Schleife (1330-1360) schreibt die vom Interface empfangenen Bytes in die Variable B(K).

SENDF -- Sende Fahrstraße
In dieses Unterprogramm wird direkt vom Fahrplan verzweigt, die Fahrstraßendaten werden dem FS-Feld entnommen (siehe INIT-SENDF).

LOKSS -- Lok schnell Stopp
SOLL- und IST-Wert einer Lok wird in diesem Programmteil auf 0 (mit Sonderfunktion = 16) gesetzt. damit ohne Verzögerung angehalten wird.

KONTA -- Kontaktabfrage und Auswertung
In Zeile 1530 wird ein vom Fahrplan definierter Kontakt solange abgefragt, bis er bestätigt wird. Bei jeder Abfrage werden die Unterprogramme LOBKE und RMLE durchlaufen. Die Auswertung der Kontaktabfrage übernimmt eine definierte Funktion (siehe INIT-RMLE).

SENDM -- sende Magnetartikel
Hiermit kann eine Einzelansteuerung eines Magnetartikels (Weiche/Signal) erfolgen.

ZEIT1 -- Zeitverzögerung (ca. 50 ms)
Die Zeitverzögerung wird zum Schalten der Magnetartikel benötigt, damit diese nicht zu kurz angesprochen werden.

ZBFSA -- Zeitlich begrenzte Fahrstufen-Direktausgabe
SOLL und IST-Wert einer Lok wird entsprechend dem Fahrplan gleichgesetzt und zeitlich begrenzt (je Schleifendurchgang ca. 330 ms. siehe auch Beschreibung (LOKBE) ausgesendet. Bei Rangierfahrten und zum Betätigen der Telexkupplung ist dieses erforderlich.

FARWE -- Fahrtrichtungswechsel
SOLL- und IST-Wert einer Lok wird auf 15 (mit Sonderf. = 31) gesetzt und ohne Verzögerung gesendet.

2. Initialisierung

Die Initialisierung ist im wesentlichen in 4 Gruppen unterteilt:

LOKBE -- In den Zeilen 2040 - 2180 sind alle Werte, welche für die Lokberechnung benötigt werden, zusammengefaßt. Die Anzahl der Loks sollte 6 nicht übersteigen, da sonst die Raktionszeit bei der Kontaktabfrage zu groß wird. Die Massensimulationskonstanten MB für Beschleunigung und MV für Verzögerung müssen experimentell für die verschiedenen Loktypen ermittelt werden.
Da der Zugriff auf Lok 6 (in Zeile 2180) vom Fahrpult (Control 8ß) aus nicht immer gewährleistet ist, wird diese zur Loksteuerung nicht verwendet.
Sind mehr als 5 Loks auf der Anlage vorhanden, können die Lokadressen und MSK-Werte vom Fahrplan aus ausgetauscht werden. Dabei muß nur sichergestellt sein, daß nicht mehr als 5 Loks gleichzeitig in Betrieb sind.

RMLE -- In Zeile 2200 wird ein Wertigkeitsfeld angelegt, dieses wird für die Schalterauswertung in Zeile 2220 benötigt. Verschiedene Versuche (Zeitmessungen)haben gezeigt, daß diese Art der Schalterauswertung sehr schnell ist. Die Variablen (2230 und 2240) und das Feld (2210) sind für die Anzahl der zu lesenden Rückmeldemodule zuständig.

MAG -- Die Magnetartikeldaten für Magnetart.-gerade (Signal-grün), Magnetart.-rund (Signal-rot) und Abschalten der Magnetart.- Spule werden Variablen (2280-2300) zugewiesen.

SENDF -- Entsprechend der Magnetartikel-Anzahl wird ein Fahrstraßen-Feld (Zeile 2340) angelegt. Hierbei ist zu beachten, daß für jeden Magnetartikel 2 Felder (Mag-Stellung und Mag-Nummer) benötigt werden (Zeile 2330). Die anlagenspezifischen Fahrstraßen werden mit den READ-anweisungen in den Zeilen 2350, 2370 und 2400 aus dem DATA-Programmspeicher gelesen. Als erstes wird die Fahrstraßen-Nummer gelesen, dann die Anzahl der Magnetartikel pro Fahrstraße und zuletzt innerhalb einer FOR-NEXT-Schleife (2390-2440) der Magnetartikel selbst. Dieser setzt sich aus Nummer und Stellung zusammen. Diese

Kombination wird in den Zeilen 2410 un 2420 in eine Mag.-Nummer und in eine Mag.-Stellung zerlegt und anschließend (Zeile 2430) ins Fahrstraßen-Feld geschrieben. Beim Erkennen einer '0'in Zeile 2360 ist das einlesen der Daten ins Fahrstraßen-Feld und die Initialisierung beendet.

3. Anlagenspezifische Fahrstraßen:

Mit der DATA-Anweisung werden ab Zeile 2530 die anlagenspezifische Fahrstraßen-Daten im Programm gespeichert. Die Anzahl und Länge der Fahrstraßen darf die Werte in dem dimensionierten Fahrstraßen-Feld (2340) nicht überschreiten. Die Reihenfolge der Fahrstraßen-Daten muß genau eingehalten werden, siehe hierzu das FS-Beispiel in den Zeilen 2470 - 2510.

4. Fahrplan:

```
3000 '_____________________________FAHRPLAN 'DEMO'
3010 '
3020 SB=RE:GOSUB 1150          'ESET-MODE "EIN"
3030 GOSUB 1300:GOSUB  1300    '2*Rückmeldemodul lesen
3040 PRINT CHR$(7)             'Piepton  Fahrplan-Start
3050 SN=1:GOSUB  1390          'Fahrstraße 1 wird ausgegeben
3060  L(1,VS)=8                'In  Variable  v Lok 1  wird  FS  8
                                geschrieben
3080  KN=13:GOSUB   1520       'Hier wird  solange  gewartet  bis
                               'Kontakt 13 betätigt wird.  Durch das
                               'zyklische      Durchlaufen       des
                               'Unterprogr.  LOKBE  werden  d.  Loks  auf  ihre  Fahrstufen
                               'beschleunigt
3090  L(1,VS)=3                'In  Variable  v Lok  1  wird  Vmin 'geschrieben
3100  KN=3:GOSUB 1520           warten a.  Kontakt 3.  Lok  1  wird verzögert
3110 SS=1:GOSUB 1480           'Lok 1 wird sofort nach Betätigen von Kontakt 3 angehalten.
3120 LN=5:F=3:Z=3:Gosub 1620    'Lok    5    wir    ca.    1    Sek    mit    Fahrstufe    3
                                angesprochen.
3130  FW=5:Gosub  1650     'Lok 5 wechselt sofort die Fahrtrichtung
3140 END
```

Folgende Programmzeilen sind rechenspezifisch und müssen entsprechend angepaßt werden: 1110, 1120. 1150, 1160, 1320, 1340, 1350 und 2020.

Spielend lernen

Berufliche Ausbildung an der Modelleisenbahn

"WISSEN IST MACHT", so stand es in dicken Lettern über dem Eingang des Schulgebäudes. "NIX WISSEN = MACHT NIX" hatten ein paar Pennäler heimlich darunter geschrieben. Die Gleichung wurde zwar mathematisch richtig erweitert, doch daß die Aussage zutrifft, daran dürften selbst die jugendlichen Schreiber nicht geglaubt haben.

Immer mehr Wissen, immer mehr Informationen müssen in der heutigen Zeit von Schülern, Studenten, den Auszubildenden aber auch von den bereits im Beruf stehenden Männern und Frauen verarbeitet werden. Daß es sich dabei nicht um eine Einbahnstraße handeln darf, daß Wissen nicht nur vorhanden, sondern auch genutzt werden sollte, ist eigentlich eine Selbstverständlichkeit. Doch wir alle wissen aus eigener Erfahrung, daß nicht selten an den beruflichen oder auch privaten Erfordernissen vorbei gelehrt wird.

Die Art und Weise, wie Wissen vermittelt wird - die Didaktik also - spielt deshalb gerade in der heutigen Zeit eine wichtige Rolle. Dies gilt im besonderen Maße auch für die Berufsausbildung. Die Anforderungen an die heutigen Fachkräfte - und hier insbesondere in den Bereichen Elektronik und Datenverarbeitung - nehmen derart rapide zu, daß herkömmliche Lehrmittel wie Tafel und Kreide nicht mehr ausreichen.

Nicht nur im schulischen Bereich, sondern gerade in der beruflichen Ausbildung wurden immer wieder neue Lehr-Konzepte erarbeitet und weiterentwickelt. Gefordert wurde dies schon durch die Berufswelt selber. So sollten beispielsweise die Facharbeiter nach ihrer Ausbildung die erforderlichen Kenntnisse besitzen, ansonsten mußte der Ausbildungsplan geändert werden. Diese 'Rückkopplung' sorgte dafür, daß man heute durchaus von einem hohen Ausbildungsniveau im beruflichen Bereich sprechen kann.

Nicht Rückkopplung aus der Berufswelt, sondern ein Impuls aus der Hobbywelt war dagegen der Initiator für eine neuartige Ausbildungsmethode, die in der Bayer AG zur Zeit in der Erprobung ist. Angehende Energie- und Kommunikationselektroniker sowie Meß- und Regelmechaniker "spielen" dort mit der Modelleisenbahn. Wobei das Wort spielen sofort korrigiert werden muß. Nicht spielen, sondern "spielend lernen", so lautet die Zielrichtung. Warum dort ein solcher Weg bei der Vermittlung von technischem Know-How beschritten wird, soll in diesem Bericht etwas genauer erläutert werden.

Fortschreitende Automatisierung in der Chemie

Immer mehr Chemie-Anlagen werden heute ganz oder zumindest in Teilbereichen vollautomatisch gefahren. Moderne Prozeßleitsysteme übernehmen die Steuerung des gesamten Prozeßablaufes. Der Mensch wird so scheinbar in die Rolle des Nur-Beobachters gedrängt; tatsächlich eröffnet diese Technik jedoch die Möglichkeit, sich mehr der eigentlichen Führung und Optimierung des Produktionsprozesses zu widmen.

Viele Entscheidungen, die früher beispielsweise vom Schichtmeister zu lösen waren, wie "die Temperatur steigt über den Maximalwert an, was ist zu tun?", werden heute bereits durch den Rechner nach ausgeklügelten Programmprozeduren und Fehlerbearbeitungsroutinen herbeigeführt.

Die Mensch-Pozeß-Kommunikation hat sich durch den Einsatz von Prozeßleitsystemen stark gewandelt. Die bisher üblichen Signal-orientierten Prozeßanlagen werden mehr und mehr durch eine Informations-orientierte Prozeßführung abgelöst.

Das Fließbild und die endloslangen Meßwarten - so, wie die großen Chemie-Anlagen auf Fotos bisher immer dargestellt wurden - werden mehr und mehr durch Bildschirm und Lichtgriffel ersetzt.

Dies führt einerseits zu einer höheren Ausbeute bei besserer Qualität der Produkte und zu einer höheren Anlagensicherheit, andererseits nimmt die Komplexität der Systeme überproportional zu. Ein großes Prozeßleitsystem in der Gesamtstruktur und im Detail zu beherrschen ist nahezu unmöglich. Es ist daher immer wichtiger, neben den erforderlichen Grundkenntnissen auch das Denken in Strukturen und 'Blackboxen' zu erlernen.

Ausbildung an der Modelleisenbahn

Daß bei der Bayer AG jetzt ein sicher etwas ungewöhnlicher Weg zur Vermittlung dieser Kenntnisse eingeschlagen wurde und Lehrlinge an einer Modelleisenbahn-Anlage ausgebildet werden, hat sicher viele Gründe. Die vier wichtigsten seien hier genannt:

1. Eine Modelleisenbahn-Anlage ist gut überschaubar. Die Wirkung von eingegebenen Funktionen oder von selbst erstellten Programmen ist sofort überprüfbar. Verfahrenstechnische Anlagen sind dagegen meist von komplexerer Struktur und besitzen häufig eine hohe Zeitkonstante, d.h. die Reaktionzeit zwischen Befehl und Wirkung liegt im Sekunden- bis Minutenbereich.

2. Auch Fehler können - gewollt oder ungewollt - simuliert und deren Wirkung unmittelbar vor Augen geführt werden. Nicht von ungefähr heißt es unter Fachleuten: "Erfahrung ist die Summe aller Mißerfolge". Hier darf also problemlos Erfahrung gesammelt werden.

3. Moderne Prozeßleitsysteme sind für große Chemieanlagen ausgelegt und daher sehr teuer. Größenordnung: eine halbe bis mehrere Millionen DM. Auch große Industrieunternehmen können daher nicht viele solcher Leitsysteme in der Ausbildung einsetzen. Die Modeleisenbahnanlage bietet eine zusätzliche preiswerte Alternative zu den vorhandenen Systemen.

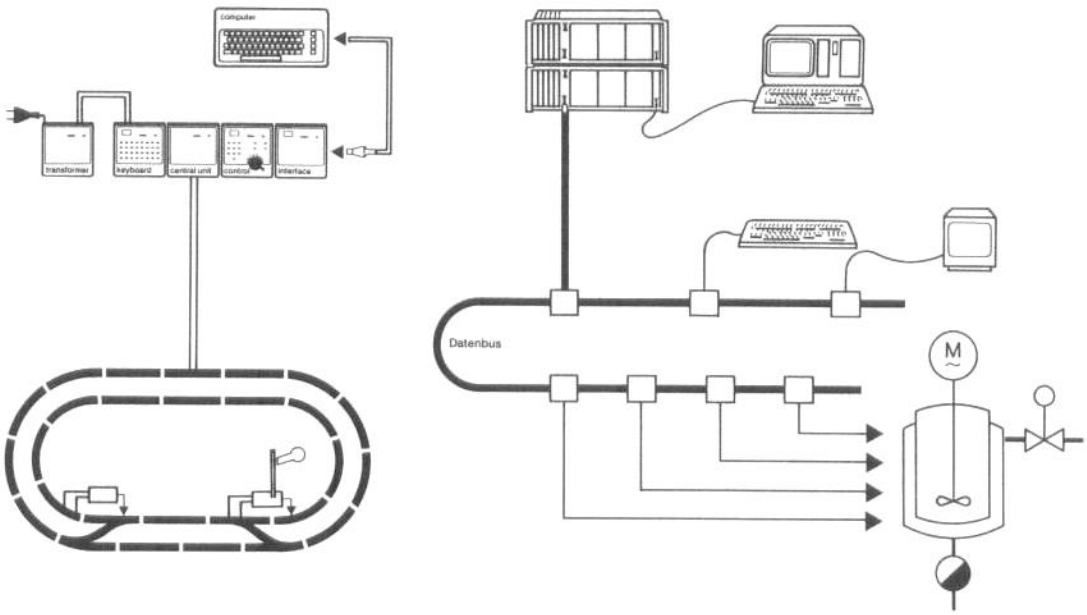
4. Die Motivation zum Lehren und Lernen am Eisenbahnmodell ist sehr groß. Was Spaß und Freude bereitet erlernt man sicherlich schneller. Es ist schon etwas anderes, ob mit einem selbsterstellten Programm ein Rangierproblem demonstriert wird, oder ob die Temperatur eines Kessels zu ändern oder zu überwachen ist.

Parallelen zur Praxis

Kritiker werden nun anmerken, daß eine chemische Produktionsanlage sicher etwas anderes ist, als eine Modelleisenbahn-Anlage. Dies stimmt natürlich, doch das oben bereits angesprochene Denken in Strukturen und 'Blackboxen' läßt sich sehrwohl auch an einem - ja, man kann durchaus sagen hochentwickelten - Spielzeug üben. Darüberhinaus lassen sich deutliche Analogien zwischen der Digital-Eisenbahn und einer verfahrenstechnischen Chemieanlage herstellen. Einige seien hier genannt:

- Die Übertragung der Daten erfolgt bei beiden Systemen auf ähnliche Weise. Bei Märklin werden alle Steuersignale über zwei Leitungen (Schienen) seriell zu den Unterstationen (Loks, Weichen, Signale) geleitet. Bei modernen Leitsystemen erfolgt die Übertragung ebenfalls seriell, wenn auch - wegen der großen zu verarbeitenden Datenmenge - über mehrere Leitungen.

- Die Steuerung der Anlage kann sowohl von Hand, als auch über Rechner erfolgen. Befehlsdaten können ausgegeben, Rückmeldungen eingelesen werden. Dies alles läuft im Echtzeitbetrieb und mit unterschiedlichem Schwierigkeitsgrad.



- Simulieren die Gleise der Modelleisenbahn beispielsweise die Rohrleitungen einer Chemieanlage, so stellen die Züge quasi den Produktstrom dar. Dieser läßt sich in der Geschwindigkeit ändern (wie die Lokgeschwindigkeit), absperren (durch Signale) oder umleiten (mittels Weichen).

- Sogar für die Rezeptur eines Produktionsprozesses gibt es ein Adäquat: der Fahrplan mit seinen Festlegungen für die zeitliche und örtliche Position der Züge.

Diese Analogien zu erkennen und in entsprechenden Programmen darzustellen gehört mit zum Ausbildungsprogramm. So lassen sich in der Ausbildungsstufe 'Spezielle Problemlösungen' durchaus interessante Aufgaben aus der Vefahrenstechnik simulieren.

Modelleisenbahn	Produktionsanlage
Modelleisenbahnanlage	Verfahrenstechnischer Prozeß
Gleise	Rohrleitungen für Produkt
Züge	Produktstrom
Signale	Absperrorgane
Weichen	Mehrwege-Ventile
Fahrplan	Rezeptur
Kontaktgleis	Signalgeber
Fahrgeschwindigkeit	Durchfluß
Abstellgleise	Vorratsbehälter
Gleisbild	Fließbild der Anlage

Die Produktstromsteuerung mittels Weichen und Signalen ist dabei die einfachste Aufgabe. Schwieriger wird da schon die Behandlung von Mischungs-und Folgesteuerungen, die mit Rangierprogrammen der Modelleisenbahn simuliert werden können. Den höchsten Schwierigkeitsgrad besitzen regelungstechnische Vorgänge. Diese können an der Modellanlage unter Umständen sogar schwerer zu realisieren sein, als in einer echten Produktionsanlage. Während die meisten Regelstrecken in der Chemie große Zeitkonstanten besitzen - je nach Branche handelt es sich bei 70 bis 80% der Regelkreise um Temperatur- und Füllstandsregelungen - so reagiert die Eisenbahn erheblich schneller. Die ansonsten erwünschte sofortige Reaktion der Anlage, kann dabei den jungen Programmierer schon ganz schön in Zeitdruck und damit ins Schwitzen bringen.

Steuern oder Regeln ?

Der Unterschied zwischen Steuern und Regeln sei hier einmal am Vergleich zwischen Chemie-Anlage und Eisenbahn-Modell aufgezeigt. Als Chemie-Beispiel sei das Füllen eines Behälters gewählt. Der Durchfluß der einfließenden Flüssigkeit soll also verändert werden.

Die einfachste und preiswerteste Lösung wäre der Einbau eines Handventils, mit dem der Zulauf der Flüssigkeit mehr oder weniger gut eingestellt werden kann. Hier haben wir es mit einer 'Hand-Steuerung' zu tun. Leider haben die preiswerten Lösungen in den meisten Fällen doch gewisse Nachteile. Die Handbedienung des Wassereinlaufs ist beim Füllen der eigenen Badewanne sicher kein Problem, in Produktionsanlagen jedoch nicht mehr vertretbar.

Wird dagegen die Anlage mit einem elektrisch oder pneumatisch angesteuerten Regelventil ausgerüstet, so kann auch vom Rechner aus die durchfließende Menge beeinflußt werden. Wir haben es jedoch hier nicht mit einer Regelung, sondern mit einer klassischen 'Rechner-Steuerung' zu tun. Die Begründung ist einfach. Ändert sich beispielsweise der Vordruck der Flüssigkeit, so führt dies ebenfalls zu einer Änderung der durchfließenden Menge. Bei exakt gleicher Zulaufzeit und exakt gleicher Stellung (Öffnung) des Ventils ist die in den Kessel geflossene Menge eben nicht gleich groß.

Reproduzierbare Werte erhält man nur dann, wenn eine echte 'Regelung' vorgenommen wird. Dazu gehört zunächst einmal, daß das zu regelnde Medium gemessen wird. Dieser Istwert wird dann mit dem vorgegebenen Sollwert verglichen und die Abweichung festgestellt. Nach bestimmten Regelalgorithmen (P, PI, PID-Verhalten) wird aus dieser Abweichung und der Änderungssgeschwindigkeit eine Änderung des Ausgangssignals vorgenommen. Auf unseren Kesselzulauf bezogen bedeutet dies, daß der Durchfluß der Flüssigkeit zunächst mit einem Durchflußmesser bestimmt werden muß. Dieser Istwert wird dem Rechner zugeführt, der ihn mit der Sollwertvorgabe vergleicht und der Abweichung entsprechend eine Änderung der Ventilstellung vornimmt.

Auch bei der digitalen Modelleisenbahn lassen sich diese Unterschiede gut demonstrieren. Die 'Hand-Steuerung' bedarf wohl keiner weiteren Erklärung. Bei der 'Rechner-Steuerung' wird die gewünschte Geschwindigkeit durch das Programm bestimmt und per Digitalkommando der Lokomotive übermittelt. Ob die tatsächliche Geschwindigkeit der Lok dieser Vorgabe entspricht, ist jedoch nur dann festzustellen, wenn die Geschwindigkeit gemessen wird. Nicht nur der Typ der Lokomotive, sondern auch die angehängte Last und die Steigung der Strecke sorgen für große Unterschiede trotz gleicher Fahrstufe.

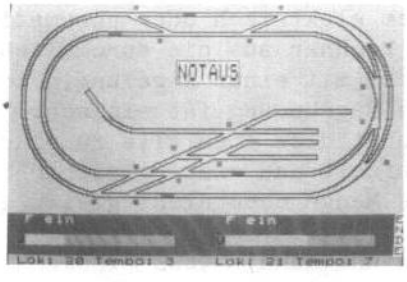
Die Messung der Fahrgeschwindigkeit ist also erforderlich. Dies kann bei der Digital-Anlage über sogenannte Rückmelde-Module erfolgen. Die Zeit, die eine Lok für das Durchfahren einer festgelegten Strecke benötigt, ist ein Maß für die Geschwindigkeit. (Das Verfahren dürfte manchem Autofahrer ebenfalls bekannt, wenn nicht sogar in schlechter Erinnerung sein!). Rückmelde-Module am Anfang und Ende der Meßstrecke melden das Überfahren einer Lok zum Rechner. Dieser berechnet dann die Geschwindigkeit und ändert gegebenenfalls die Fahrstufe. Welche Lok die Meßstrecke passiert hat, läßt sich jedoch bei dieser relativ einfachen Methode nicht feststellen. Hier ist das Gedächtnis und die Kombinationsfähigkeit des Rechners gefordert.

Da regelungstechnische Begriffe angesprochen wurden, sei noch ein Hinweis auf die heute üblichen Fachausdrücke angebracht. Meßwertaufnehmer, hier also das Kontaktgleis oder Rückmeldemodul, werden fachspezifisch 'Sensoren' genannt. Bausteine, die eine Regelstrecke aktiv beeinflussen, also das Ventil oder bei der Modellbahn die Motoren der Lokomotiven, nennt man in der Prozeßleittechnik 'Aktoren'. Sollte in einem Fachbuch der Begriff 'Regelgröße' auftauchen, so handelt es sich um den Istwert, bei 'Führungsgröße' schlichtweg um den Sollwert des Regelkreises. Soweit diese Exkursion in die Regelungstechnik.

Rechner und Peripherie

Ein wichtiger Aspekt bei der Ausbildung an der Modelleisenbahnanlage ist auch der Umgang und die Bedienung von Rechner und Rechnerperipherie. Auch hier sind wieder starke Parallelen zum großen Bruder Prozeßleitsystem festzustellen. Aufbau, Struktur und Hierarchie der Rechnersysteme sind einander ähnlich, Unterschiede liegen nur in der Größenordnung und in der Verarbeitungsgeschwindigkeit.

Bei der Bedienung des Rechners wird ebenfalls auf eine Analogie zur späteren Praxis Wert gelegt. So ist die Benutzung eines Lichtgriffels an einem Grafikbildschirm obligatorisch, da bei der Mehrheit der Prozeßleitsysteme ebenfalls der Lichtgriffel zur Auswahl und Ansteuerung einzelner Regelkreise eingesetzt wird. Gute Erfahrungen wurden bei der Modelleisenbahnansteuerung mit dem THOMSON Rechner TO7 gemacht. Dieser ist standardmäßig mit einem Lichtgriffel ausgerüstet und besitzt auch für die grafische Gestaltung eine ansprechende Software. Leider hat sich dieser Rechnertyp auf dem deutschen Markt nicht durchsetzen können. Die Lichtgriffelsteuerung bei anderen Homecomputern ist schon der Auflösung wegen nicht optimal. Zur Zeit werden Versuche mit IBM-kompatiblen Rechnern (mit Grafikkarte und Lichtgriffelsteuerung) durchgeführt.



Wie sehr auf dem Computersektor eine gegenseitige Beeinflussung von Hobby und Arbeitswelt stattfindet, läßt sich anhand des Joysticks zeigen. Einst als Spielzeug und Bediengerät für Video- und Computerspiele entwickelt, wird dieser mittlerweile auch bei der Verfahrensführung chemischer Prozesse eingesetzt. Dort sowohl beim Verschieben, bzw. Durchfahren des Fließbildes auf dem Grafikbildschirm, als auch beim Einstellen der Prozeßparameter (Arbeitspunkt) im dreidimensionalen Verfahrensraum.

Ob die Maus, die ursprünglich für die CAD-Anwendung (Computer Aided Design) entwickelt wurde und heute bereits bei vielen Homecomputern zur Standardausrüstung gehört, auch ihren Einzug in die Meßwarten halten wird, ist jedoch fraglich.

Die Programmerstellung

Die digitale Modelleisenbahnanlage bietet den großen Vorteil, bereits mit wenig Programmieraufwand schon sichtbare Erfolge auf den Schienen herbeizuführen. Ein kleines Programm mit ein paar BASIC-Befehlen kann bereits eine Lok langsam anfahren und nach

einiger Zeit wieder schrittweise abbremsen lassen. Das Zusammenspiel von Computer und Eisenbahnelektronik wird so gut demonstriert. Das Ziel dieser Ausbildungsstufe ist jedoch das strukturierte Programmieren. Also weg vom "Spaghetti-BASIC-Programm" und hin zu Unterprogrammen und Prozeduren zur Lösung von Teilaufgaben. Dies läßt sich sicher auch in BASIC realisieren, angestrebt wird jedoch das Arbeiten mit PASCAL. Diese Sprache zwingt zu einer Strukturierung der Programme und wird auch häufig in der Industrie zur Lösung technischer Probleme eingesetzt.

Anhand eines kleinen Programmbeispiels soll der "Sprachunterschied" zwischen BASIC, PASCAL und der Assemblerprogrammierung aufgezeigt werden. Alle drei Programme haben die gleiche Aufgabenstellung: Langsames schrittweises Anfahren einer Lokomotive bis zur Maximalgeschwindigkeit, danach Beibehaltung dieser Geschwindigkeit für eine vorgegebene Zeitdauer, anschließend schrittweise Verringerung der Geschwindigkeit bis zum Stillstand. Man spricht auch von einem trapezförmigen Geschwindigkeitsprofil. Eine einfache Aufgabe also, die gut nachzuvollziehen ist.

Die drei Programmbeispiele sind hier bewußt kurz gehalten. Ebenso soll auf eine detaillierte Beschreibung verzichtet werden, der Sinn liegt mehr in der Gegenüberstellung der Programmstrukturen. Alle Programme sind jedoch voll funktionstüchtig. Entwickelt wurden sie auf und für den EPSON Handheld-Computer PX8, sie sind jedoch ohne Schwierigkeiten auf andere Rechner übertragbar. Der Assembler ist für 8080-CPUs programmiert und somit auf allen CP/M-Rechnern lauffähig.

Auf die Einstellung der Schnittstellenparameter wurde bewußt verzichtet, da diese stark vom verwendeten Rechnertyp abhängen. Spezifische Schnittstellenfestlegungen müssen deshalb noch in die Programme eingebunden werden oder vor der Programmausführung mittels spezieller Konfigurationsprogramme (CONFIG, DEVICE o. ä.) vorgenommen werden.

Bus-Strukturen

Anders als im schulischen Bereich - in dem bereits auch mit Digital-Eisenbahn und Computer gearbeitet wird - spielt bei der beruflichen Ausbildung neben dem Zusammenspiel von Rechner und Anlage auch der Aufbau der Hardware eine große Rolle. Meßinstrumente und Meßwertaufnehmer werden immer "intelligenter" und ermöglichen so eine Kommunikation untereinander. Der Austausch von Daten und Informationen erfolgt häufig über sogenannte Bus-Leitungen. Man unterscheidet vereinfacht gesehen zwischen der parallelen und der seriellen Bus-Übertragung.

Ein Hinweis dazu. Auch bei der Druckeransteuerung sind beide Übertragungsmodi möglich. Man spricht von der parallelen CENTRONICS- und von der seriellen RS232-Schnittstelle.

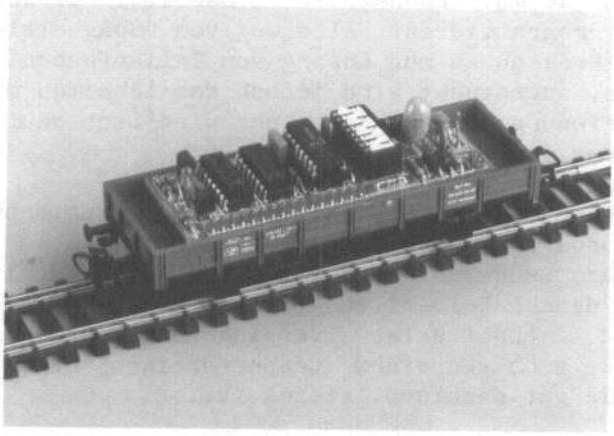
Besonders die serielle Datenübertragung besitzt ein sehr komplexes Übertragungsprotokoll, das schwer überschaubar ist. Da die Datenbits mit hoher Geschwindigkeit nacheinander übertragen werden, sind herkömmliche statische Meßmethoden nicht anwendbar. Die Märklin digital-H0 stellt somit ein gutes Objekt dar, an dem die Abläufe und Prozeduren der seriellen Datenübertragung beobachtet, dekodiert und simuliert werden können.

Leider hat sich die Firma Märklin für die Kodierung den nicht sehr geläufigen Trinär-Code gewählt, der Parallelen zu den sonst üblichen Kodierungen erschwert. Diese Kodierungsart war jedoch erforderlich, weil - so läßt sich jedenfalls vermuten - die Signalspannung gleichzeitig auch Versorgungsspannung ist.

Die Bus-Struktur der modernen Prozeßleitsysteme ist jedoch noch komplexer. Es handelt sich meist um firmenspezifische Bus-Systeme, die einerseits den Datenaustausch zwischen den internen Baugruppen des Prozeßleitsystems vornehmen, andererseits die Kommunikation mit den Geräten in der Anlage durchführen.

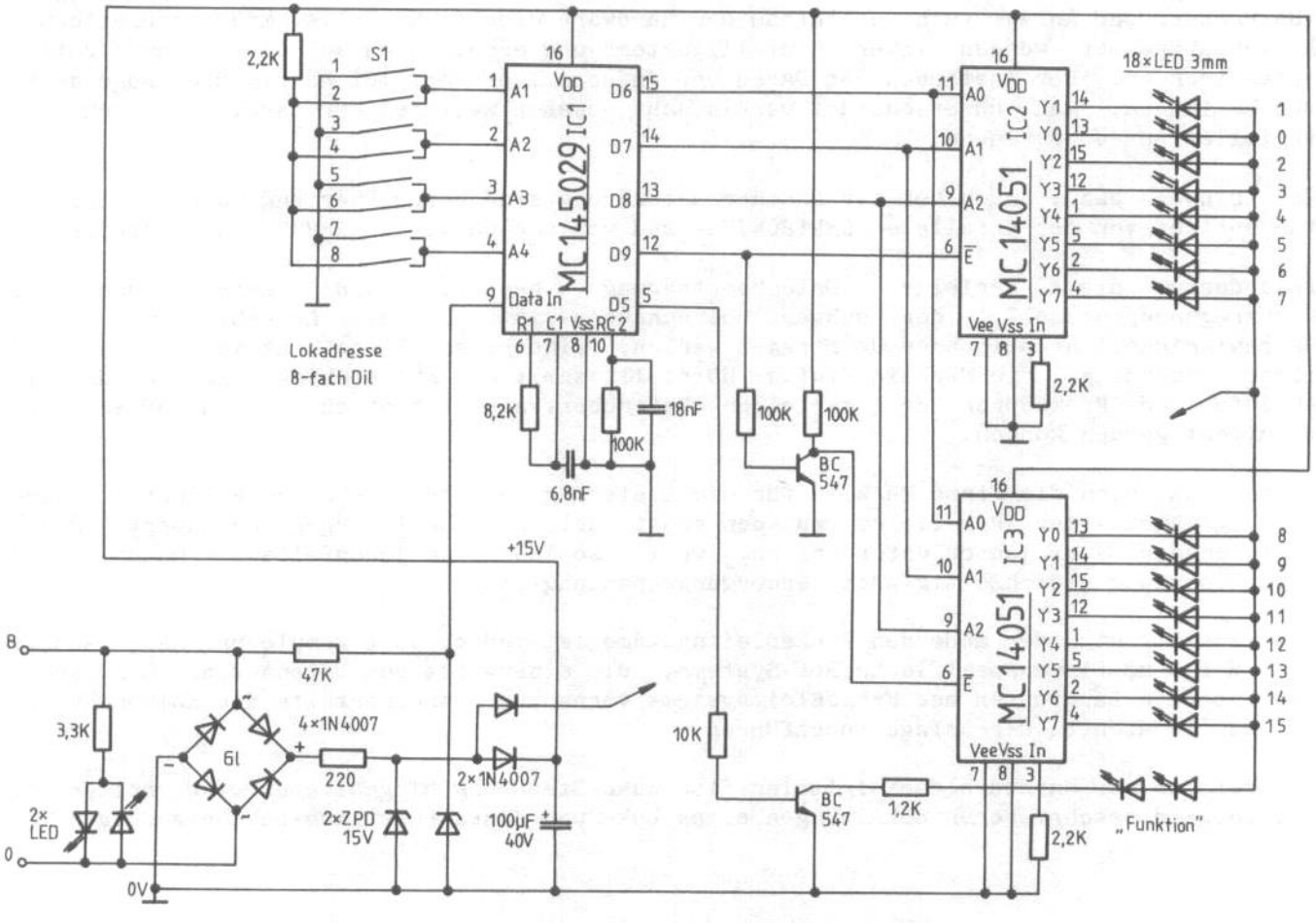
Wie weit in die Hardware der digitalen Eisenbahn-Steuerung eingestiegen wird, sollen die nachfolgend beschriebenen Schaltungen eines Lok- und eines Funktions-Dekoders zeigen.

Der Lok-Dekoder



Er ist nicht nur hübsch anzuschauen, er funktioniert auch einwandfrei, der kleine Güterwagen mit dem Elektronikaufbau. Um die Funktion der Bus-Struktur praxisnah nachzuvollziehen, wurde ein Lok-Dekoder für die Märlin Digital-H0 'zurückentwickelt'. Dies hört sich ein wenig seltsam an, ist jedoch leicht erklärbar. Während der Hersteller verständlicherweise versucht, immer kleinere Dekodermodule zu entwickeln (z.B. in Form von Einzelchips zum Einbau in kleinste Modelle), wurde hier von Auszubildenden eine Schaltung mit diskreten Bauteilen erstellt. Dies, um einerseits die Funktion der Schaltung besser zu demonstrieren, andererseits, weil so marktgängige Bauteile eingesetzt werden konnten.

Die Schaltung verhält sich wie der Märklin Lok-Dekoder, nur wird anstelle des Motorantriebes die ausgegebene Fahrstufe mit Hilfe einer Leuchtdiodenreihe angezeigt. Die Adressencodierung erfolgt wie bei den Lokomotiven über 8polige DIL-Schalter und auch bei den Schalterstellungen wurde sich an die durch Märklin festgelegten Vorgaben gehalten.



Die Schaltung bedarf keiner umfangreichen Erläuterung. Die Adressenerkennung und Datenspeicherung wird durch IC1 MC145029 vorgenommen, einem Spezialbaustein für die trinäre Datenübertragung. Das RC-Netzwerk an den Pins 6, 7, 8 und 10 ist für die Festlegung der Oszillatorfrequenz des Bausteins wichtig. Die angegebenen Werte sind einzuhalten, da die Empfangsfrequenz mit der Sendefrequenz des Märklin-Transformers logischerweise übereinstimmen muß. Auch bei anderen seriellen Datenübertragungen ist dies ebenfalls eine notwendige Voraussetzung.

Bei den beiden MC14051 handelt es sich um Datendemultiplexer, die hier als Dekodierer und Treiber zur Ansteuerung der Leuchtdioden eingesetzt wurden. Die beiden 2,2 kOhm Widerstände an Pin 3 bestimmen den LED-Strom und damit deren Helligkeit. Die rechts neben den Leuchtdioden angegebenen Zahlen geben die Fahrstufe an. Hier ist zu bemerken, daß die Fahrstufe 1 dem Rückwärtsimpuls entspricht (Der Fachmann weiß es besser: Fahrtrichtungänderungsimpuls muß es heißen). Auch dies eine Konzession an das trinäre Märklin-Konzept.

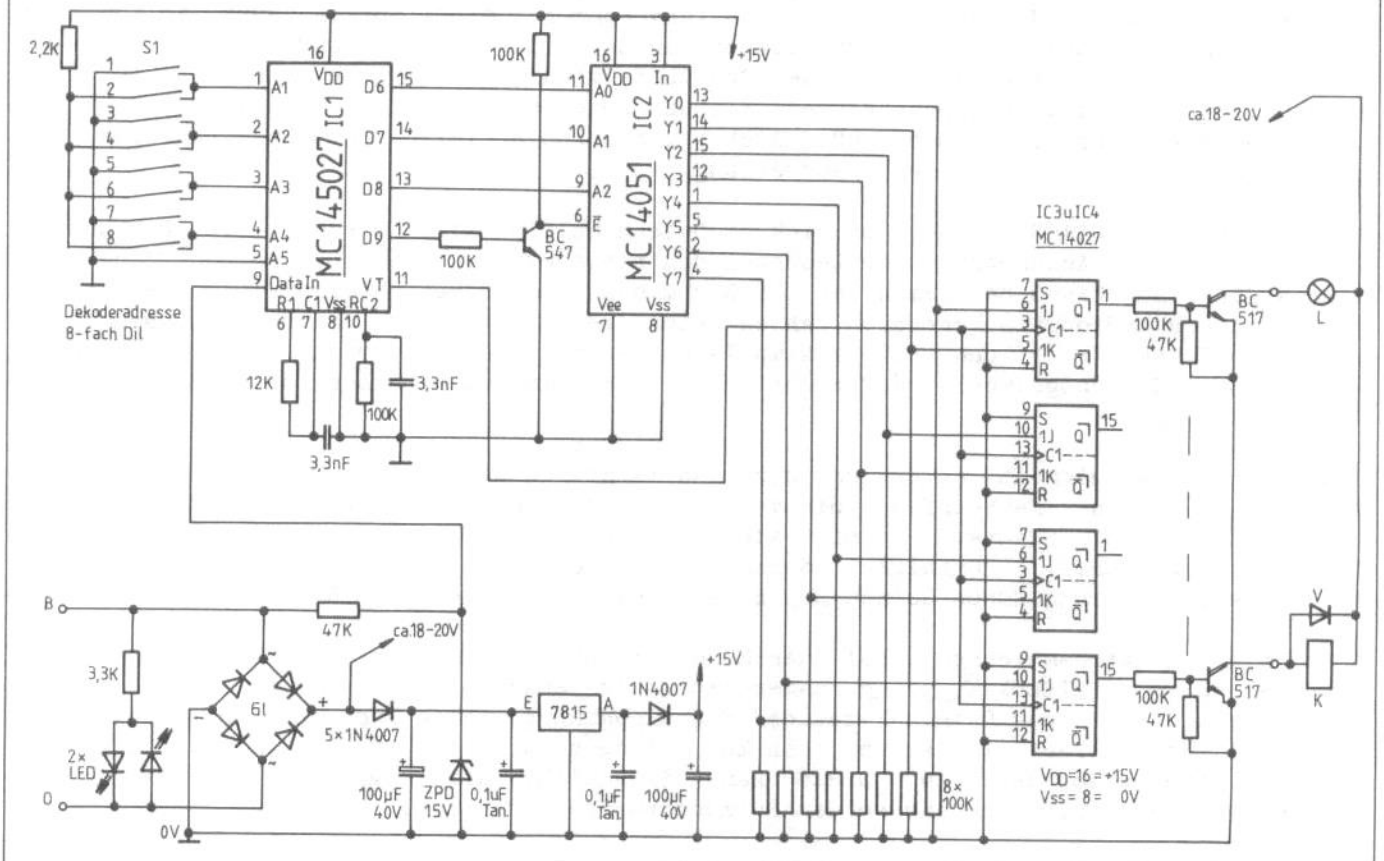
Die Fahrgeschwindigkeit entspricht also den 14 Stufen 0, 2, 3 ..... 15; nicht zu verwechseln mit den Fahrstufen, die vom Rechner ans Interface gesendet werden müssen. Hier ist für die Fahrstufen die Kodierung 0, 1, 2 .... 14 festgelegt, der Fahrtrichtungswechsel erfolgt durch Ausgabe der Information 15, die Funktionseinschaltung durch Setzen des 5. Bits (also Addition von 16).

Auch die Funktionsmeldung - bei der Modelleisenbahn zum Ein- und Ausschalten des Lichtes, bzw. der TELEX-Kupplung ausgenutzt - wird erkannt und angezeigt. Gespeist wird die Schaltung wie das Original aus der Signalspannung, also über die Schienen. Der Elektronik-Wagen läßt sich so an jeden Zug anhängen und zeigt stetig die Fahrstufe der eingestellten Lokadresse an. Diese kann die gleiche sein, wie die der ziehenden Lokomotive, aber auch jede andere der 80 zugelassenen Adressen.

Der Funktions-Dekoder, oder der Koch schaut aus dem Fenster

Der Appetit kommt bekanntlich über dem Essen, und so werden bei der digital steuerbaren Modellbahn Wünsche geweckt, über die man früher nur den Kopf schütteln konnte. Heute sind diese von Modellbauspezialisten - dank der Digital-Technik - durchaus realisierbar. Man stelle sich einmal vor, daß die Lok - vom Stellpult aus gesteuert - vor der Tunneleinfahrt pfeift, die Beleuchtung des Toilettenabteils ein- und wieder ausgeschaltet wird oder als besonderer Gag, daß der Koch das Fenster öffnet, um sich die Gegend anzuschauen. Dies alles selbstverständlich während der Fahrt und ohne großen Elektronikaufwand.

Möglich ist beispielsweise ebenfalls die Steuerung eines fahrbaren Kranwagens oder das Ausfahren der Oberleitungs-Stromabnehmer. Auch diese Ansteuerung wird durch das Stellpult oder - wie es heute ja genannt wird - das Keyboard der Anlage ausgelöst.



Auch diese Schaltung wurde im Ausbildungsbereich entwickelt und ausgetestet. Für "Digital-Eisenbahner" mit etwas Elektronikerfahrung empfiehlt sich durchaus der Nachbau.

Bei der Schaltung handelt es sich um einen diskret aufgebauten Weichen- und Signal-Dekodierer mit zusätzlichem Speicherverhalten. Die Adressenerkennung erfolgt ähnlich wie beim Lok-Dekoder, jedoch mit einem anderen Trinärdekoder MC145027. Märklin verwendet bekanntlich für die Weichen- und Lok-Ansteuerung unterschiedliche Übertragungsgeschwindigkeiten. Für die Lokdaten: 19 kHz, für Weichen und Signale: 38 kHz. Aus diesem Grund ist auch die Resonanzfrequenz dieses Funktions-Dekoders eine andere. Nach der Dekodierung durch den MC14051 wird die übertragene Funktion in den JK-Flip-Flops IC1 und IC2 (MC14027) gespeichert. Diese Daten-Übernahme erfolgt bei positivem Signal VT ("Valid Transmission", Pin 11 von IC1). Der Transistor BC547 dient der Invertierung des vierten Datenbits (D9 von IC1). Somit werden die ersten vier Kanäle der eingestellten Dekoderadresse angesprochen. Sollten die Kanäle 5 bis 8 angesteuert werden, so ist VT direkt mit dem Enable-Eingang zu verbinden.

Die nachfolgende Signalverarbeitung schließlich ist von der Anwendung abhängig, und muß durch den Anwender entsprechend angepaßt werden. Bei der vorliegenden Schaltung sind Ausgangstransistoren vorgeschlagen, mit denen Leuchtdioden, Lampen, Relais oder auch Kleinmotoren angesteuert werden können.

Auch hier erfolgt der Abgriff der Signal- und Speisespannung über die Schienen. Da mehr Strom für die Ansteuerung der Lampen, Relais oder sonstige Verbraucher benötigt wird, wurde diesmal ein Spannungsregler 7815 eingesetzt. Die beiden Eingangs-Leuchtdioden zeigen den einwandfreien Betriebszustand an.

Auch die Ausgänge des Lok-Dekoders lassen sich selbstverständlich über Transistoren für andere Belastungen verstärken. Die Arbeitsweise ist jedoch eine andere, als die des Funktions-Dekoders. Zur besseren Veranschaulichung sei der Unterschied zwischen Lok-Dekoder und Funktions-Dekoder noch einmal herausgestellt. Der Lok-Dekoder gibt ein Einzelsignal aus, das der Fahrstufe der angewählten Lokomotive entspricht. Es wird jeweils nur ein Kanal (Leuchtdiode), eingeschaltet. Bei Änderung der Lokadresse wird diese Information beibehalten.

Der Funktions-Dekoder stellt vier Einzelkanäle zur Verfügung, die jeweils unabhängig voneinander ein- und ausgeschaltet werden können. Die Bedienung erfolgt über das Weichen- oder Signal-Keyboard. Die Ein- und Aus-Signale müssen vom Funktions-Dekoder gespeichert werden, da eine Zustandsspeicherung wie bei den Weichen oder Signalen fehlt.

Falls gewünscht, läßt sich der Funktions-Dekoder auf 8 Kanäle erweitern. Dies ist realisierbar durch Einfügen eines weiteren Dekoders MC14051 und zwei weiterer Flip-Flops MC14027. Der Anschluß erfolgt in ähnlicher Weise wie die 16 Kanalschaltung des Lok-Dekoders.

Wie diese Beispiele zeigen, ist die berufliche Ausbildung an einer Modelleisenbahn-Anlage durchaus ein Weg, komplizierte technische Sachverhalte darzustellen.

Im Frühjahr 1985 wurde erstmals das System "Märklin digital" der Öffentlichkeit vorgestellt und schon bald wurden die Analogien zu den bestehenden Prozeßleitsystemen erkannt. Dies führte zu einer spontanen, sehr intensiven Zusammenarbeit mit der Firma Märklin.

Ein Ergebnis dieser Verbindung stellt das beschriebene Ausbildungs-Konzept dar, das sich - wie oben bereits gesagt - zunächst noch in der Erprobungsphase befindet. Die ersten Ergebnisse und vor allen Dingen die Begeisterung der Jugendlichen lassen jedoch die Prognose zu, daß diese Art der Wissensvermittlung einen festen Platz im breiten Ausbildungsspektrum einnehmen wird.

Wolfgang Otternberg

```
PROGRAM Lok_Ansteuerung;
{ Demonstrationsprogramm zur Lokansteuerung in PASCAL }
{ Schnittstelle muss vor dem Start konfiguriert sein  }

TYPE  MiniString  = string[2];

CONST Zeit1       = 200;   { Zeit zwischen Fahrstufen }
      Zeit2       = 2000;  { Zeit fuer Maximalgeschw. }

VAR   Eingabe,
      Wahl        : char;
      LokAdresse,
      Fahrstufe,
      Zeitfaktor,
      FuCode,
      Error       : integer;
      Funktion    : boolean;


PROCEDURE LokAdresseHolen;
    BEGIN
      REPEAT
        ClrScr;
        GotoXY(15,3);
        write('Bitte geben Sie die Lok-Adresse ein: ');
        {$I-} readln(LokAdresse);
        GotoXY(15,5);
        write('Soll die Funktion (E)in oder (A)us geschaltet sein ? ');
        read(KBD,Eingabe) {$I+};
      UNTIL ((LokAdresse IN[1..80])AND(IOresult=0));
      GotoXY(30,7);
      IF Upcase(Eingabe)='E' THEN Funktion:=True ELSE Funktion:=False;
    END;

  PROCEDURE ZeitfaktorHolen;
    BEGIN
      REPEAT
        ClrScr;
```

```
      GotoXY(10,3);
      write('Geben Sie den Zeitfaktor ein  1=schnell  9=langsam : ');
      read(KBD,Eingabe);
      Val(Eingabe,Zeitfaktor,Error);
    UNTIL ((Zeitfaktor IN[1..9])AND(Error=0));
  END;

PROCEDURE DatenAusgeben (Daten : MiniString);
  BEGIN
    {$I-} write(AUX,Daten) {$I+};
    IF IOresult<>0 THEN
      BEGIN
        ClrScr;
        GotoXY(20,5);
        write('Fehler bei der Datenausgabe !!!');
      END;
    END;

PROCEDURE Fahren;
  BEGIN
    Fahrstufe:=0;
    IF Funktion THEN FuCode:=16 ELSE FuCode:=0;
    ClrScr;
    GotoXY(18,3);
    write('Lok-Adresse:',LokAdresse:3);
    GotoXY(50,3);
    IF Funktion THEN write('Funktion ein !')
                ELSE write('Funktion aus !');
    DatenAusgeben(chr(Fahrstufe+FuCode)+chr(LokAdresse));
    While Fahrstufe<14 DO
      BEGIN
        Delay(Zeit1*Zeitfaktor);
        Fahrstufe:=succ(Fahrstufe);
        DatenAusgeben(chr(Fahrstufe+FuCode)+chr(LokAdresse));
        GotoXY(33,6);write('Fahrstufe: ',Fahrstufe:2,'  ');
      END;
    Delay(Zeit2*Zeitfaktor);
    While Fahrstufe>0 DO
      BEGIN
        Fahrstufe:=pred(Fahrstufe);
        DatenAusgeben(chr(Fahrstufe+FuCode)+chr(LokAdresse));
        GotoXY(33,6);write('Fahrstufe: ',Fahrstufe:2,'  ');
        Delay(Zeit1*Zeitfaktor);
      END;
    DatenAusgeben(chr(0)+chr(LokAdresse)); { Funktion aus }
  END;

BEGIN  { Hauptprogramm }
  LokAdresse:=0;
  Zeitfaktor:=1;
  ClrScr;
  GotoXY(20,5);
  write('******** Demoprogramm fuer Digital H0 ********');
  Delay(2000);
  REPEAT
    ClrScr;
    GotoXY(25,1); write('A u s w a h l - M e n u e ');
    GotoXY(28,3); write('L = Lok-Adresse eingeben');
    GotoXY(28,4); write('A = Ablaufzeit aendern');
    GotoXY(28,5); write('S = Lok starten');
    GotoXY(28,6); write('X = Programm beenden');
    GotoXY(60,8); write('Ihre Wahl: ');
    read(KBD,Eingabe);
```

```
    Wahl:=Upcase(Eingabe);
      CASE Wahl OF
        'L' : LokAdresseHolen;
        'A' : ZeitfaktorHolen;
        'S' : BEGIN
                IF LokAdresse=0 THEN LokAdresseHolen;
                Fahren;
              END;
      END;
  UNTIL Wahl='X';
END.  { Hauptprogramm }
```

```
10 REM ***** Demonstration zur Lokansteuerung in BASIC *****
20 'Schnittstelle muß vor Programmstart konfiguriert sein
30 :
40 EXTERN=1                    'RS232 ist zusätzlicher Kanal
50 'EXTERN=0                   'RS232 ist Drucker-Kanal
60 CLS                         'evtl. Print Bildschirmlöschsequenz
70 INPUT "Bitte die Lokadresse eingeben (1-80):",I$
80 LA=INT(VAL(I$))
90 IF LA<1 OR LA>80 THEN CLS:GOTO 70
100 INPUT "Soll die Funktion ein oder aus sein (e/a) ";I$
110 IF I$="E" OR I$="e" THEN FU=16 ELSE FU=0
120 IF I$<>"A" AND I$<>"a" AND I$<>"E" AND I$<>"e" THEN CLS:GOTO 100
130 IF EXTERN THEN OPEN "O",#1,"COM0:"  'Schnittstelle öffnen
140 FOR N=0 TO 14
150 DAT$=CHR$(N+FU)+CHR$(LA)
160 GOSUB 600                  'Serielle Ausgabe
170 GOSUB 400                  'PAUSE 1
180 NEXT N
190 GOSUB 500                  'PAUSE 2
200 FOR N=14 TO 0 STEP-1
210 DAT$=CHR$(N+FU)+CHR$(LA)
220 GOSUB 600
230 GOSUB 400
240 NEXT N
250 FU=0                       'Funktion aus
260 DAT$=CHR$(FU)+CHR$(LA)
270 GOSUB 600
280 PRINT "Programm beendet"
290 IF EXTERN THEN CLOSE
300 END
310 :
400 REM ** Pause 1 zwischen Geschwindigkeitsänderung **
410 FOR P=0 TO 400
420 NEXT P
430 RETURN
440 :
500 REM ****** Pause 2 bei Höchstgeschwindigkeit ******
510 FOR P=0 TO 2000
520 NEXT P
530 :
600 REM ********** Datenausgabe ********************
610 IF EXTERN THEN PRINT#1,DAT$ ELSE LPRINT DAT$
620 RETURN
```

```
;Demonstration der Lokansteuerung fuer 8080 ASSEMBLER
;Märklin Digital H0
;Serielle Ausgabe für CP/M Systeme
;RS 232  2400 Baud, 8 Datenbits, None Parity, 2 Stopbits

BDOS    EQU     5
PUNCH   EQU     4

        ORG     100H

START:  LXI     H,LA      ;Lokadresse laden
        MOV     E,M
        LXI     H,FU      ;Funktion ein/aus holen
        MOV     D,M
        JMP     LOOP1A

LOOP1:  POP     A         ;Lok langsam hochfahren
        INR     A         ;Tempo + 1
        CPI     15        ;Höchstgeschwindigkeit erreicht?
        JZ      LOOP2
LOOP1A: PUSH    A
        ORA     D         ;Funktion dazu
        CALL    SIOOUT
        MOV     A,E       ;Lokadresse
        CALL    SIOOUT
        CALL    PAUSE1
        JMP     LOOP1

LOOP2:  PUSH    A         ;Lok langsamer
        CALL    PAUSE2
LOOP2A: POP     A
        DCR     A         ;Tempo - 1
        JZ      LOOP3     ;Geschwindigkeit = 0 ?
        PUSH    A
        ORA     D
        CALL    SIOOUT
        MOV     A,E
        CALL    SIOOUT
        CALL    PAUSE1
        JMP     LOOP2A

LOOP3:  CALL    SIOOUT    ;Stillstand u. Funktion aus
        MOV     A,E
        CALL    SIOOUT
        JMP     0         ;Ende

;****   UNTERROUTINEN     ****

PAUSE2: MVI     L,0FH     ;Zeit bei Höchstgeschwindigkeit
        JMP     PAUSE
PAUSE1: MVI     L,2       ;Zeit zwischen Tempoänderung
PAUSE:  PUSH    D
        PUSH    A
LOOP4:  LXI     B,0FFFFH  ;Zeitschleife
LOOP4A: DCX     B
        MOV     A,C
        ORA     B
        JNZ     LOOP4A
        DCR     L
        JNZ     LOOP4
        POP     A
        POP     D
        RET
```

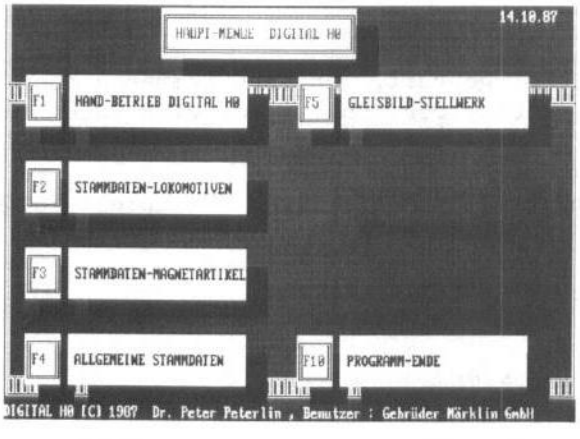
```
SIOOUT: PUSH    D       ;Start Serielle Ausgabe
        PUSH    A       ;      über Puncher
        MOV     E,A     ;      im CP/M-System
        MVI     C,PUNCH
        CALL    BDOS
        POP     A
        POP     D
        RET

;*****    VARIABLE    *****

LA:     DB      21      ;Lokadresse (1-80),hier 21
FU:     DB      16      ;Funktion 16=ein,0=aus
```

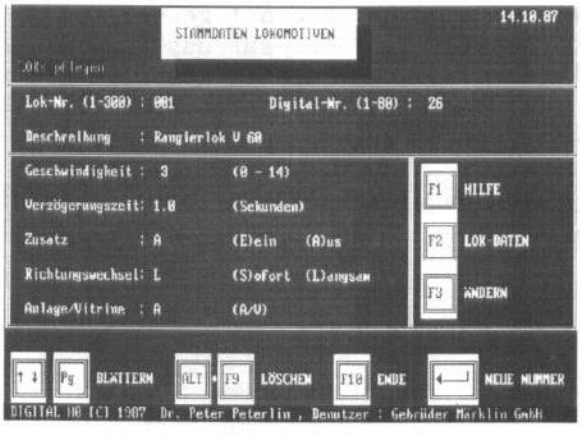
Ein interaktives Universalprogramm

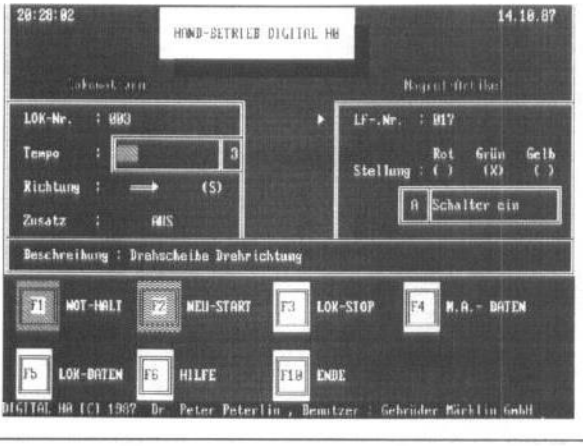
Zur Steuerung von Märklin Digital HO, einschließlich der Darstellung von Gleisbildern auf dem Bildschirm hat Dr. Peter Peterlin ein Programmsystem entwickelt, das die universelle Kontrolle einer Digital-HO-Anlage mit einem IBM PC kompatiblen Rechner gestattet. Was bis jetzt noch fehlt ist die Integration der Rückmeldemodule. Ziel ist es, abgesichert durch Rückmeldemodule, Fahrpläne erstellen und abfahren zu können.

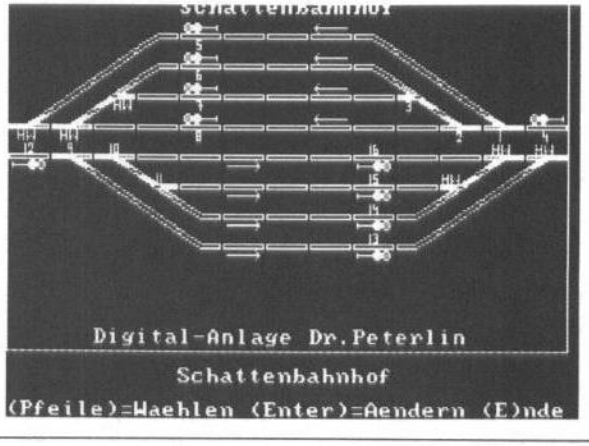
Mit dem Programm werden die Stammdaten, also alle Informationen über Triebfahrzeuge und Magnetartikel erfaßt und in einer Liste geführt. Mit einem Gleisbildeditor lassen sich einzelne Bahnhofsbereiche auf dem Bildschirm darstellen. Die Auflösung am CGA-Bildschirm reicht nicht zur Darstellung komplexer Gesamtanlagen aus.

Menügeführt lassen sich jetzt schon Lokomotiven steuern und Magnetartikel stellen. Die Bildschirmkopien geben auch Anregungen um für eigene Programme eine übersichtliche Benutzeroberfläche zu entwickeln. Wer in Turbo-Pascal programmiert kann dazu auf die Pascal Library der CHIP Special Reihe zurückgreifen. Für Turbo-BASIC gibt es eine Fensterverwaltung im CHIP Special Turbo-BASIC Heft.



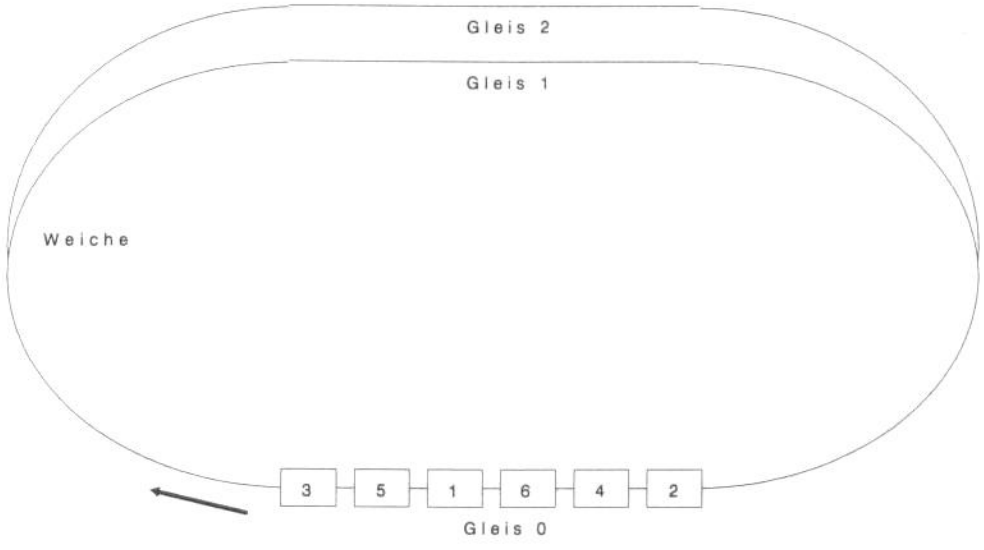






Loksort

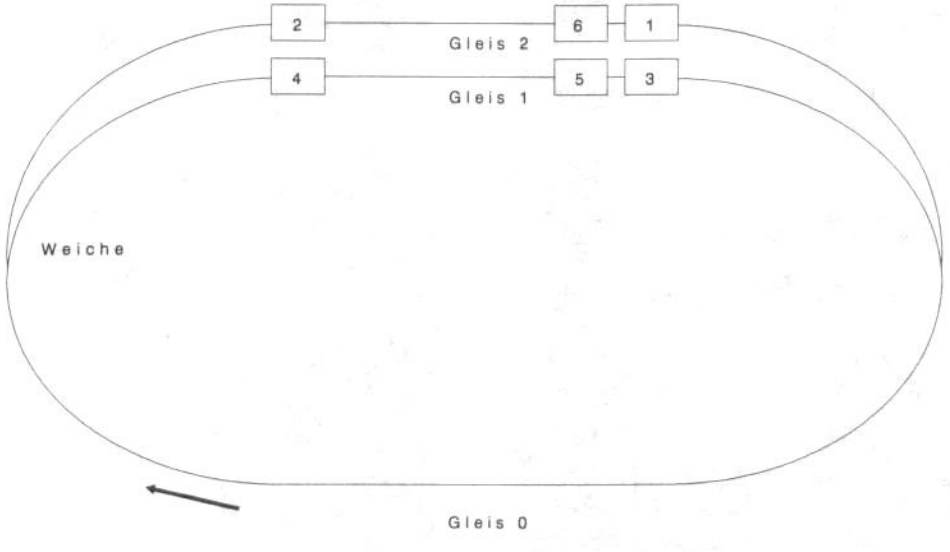
Das Programm **LokSort** demonstriert einen besonders interessanten Algorithmus, der hauptsächlich zur Sortierung großer Dateien eingesetzt wird. In unserer Eisenbahnsimulation wird eine solche Datei als Foge mehrere Lokomotiven dargestellt. Diese befinden sich ausgangs auf dem Hauptgleis (Gleis 0) in beliebiger Reihenfolge. Zusätzlich zu diesem Hauptgleis stehen zwei Rangiergleise zur Verfügung, auf die mit der Weiche umgestellt werden kann. Die einzelnen Lokomotiven fahren grundsätzlich nur in der angezeigten Fahrrichtung.



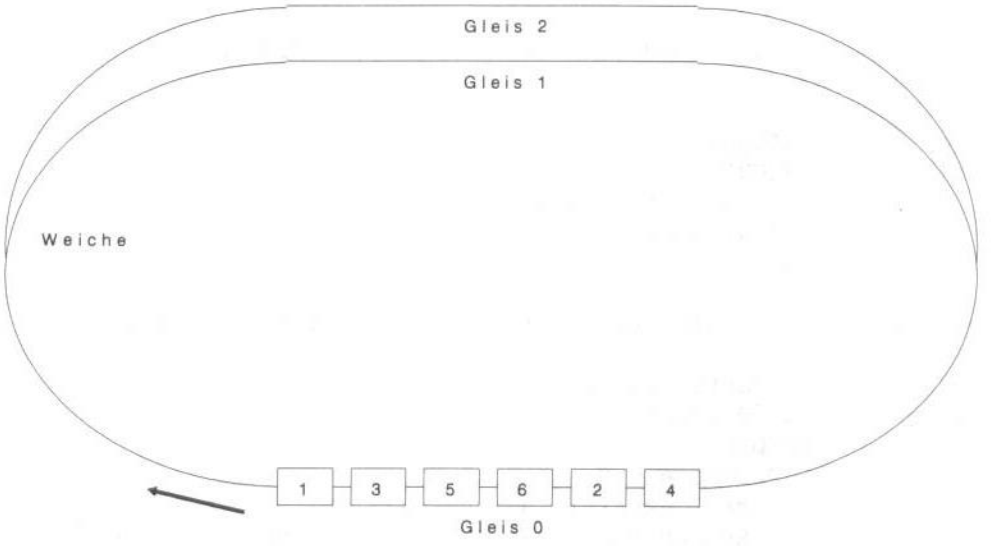
Die Sortierung erfolgt nun in mehreren Phasen:

1.Phase:
Die Weiche wird auf Gleis 1 gestellt.
Die vorderste Lok fährt auf das Rangiergleis.
Bei der jeweils vordersten Lok wird nun immer erst geprüft, ob ihre Nummer größer ist als die der zuletzt gefahrenen. Ist dies nicht der Fall, so wird die Weiche umgeschaltet. Erst dann darf die Lok fahren.

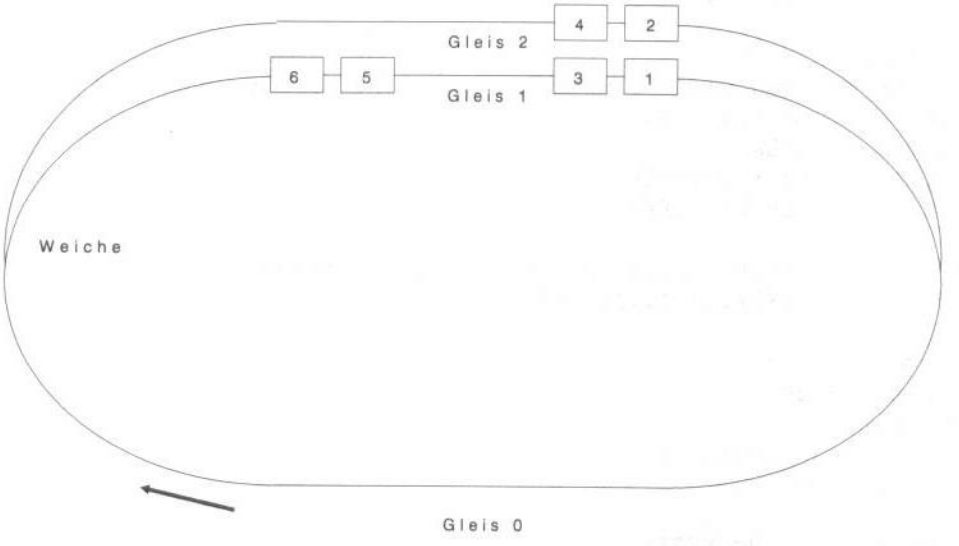
Wenn das Hauptgleis leer ist, haben die Rangiergleise in unserem Beispiel folgende Belegung.



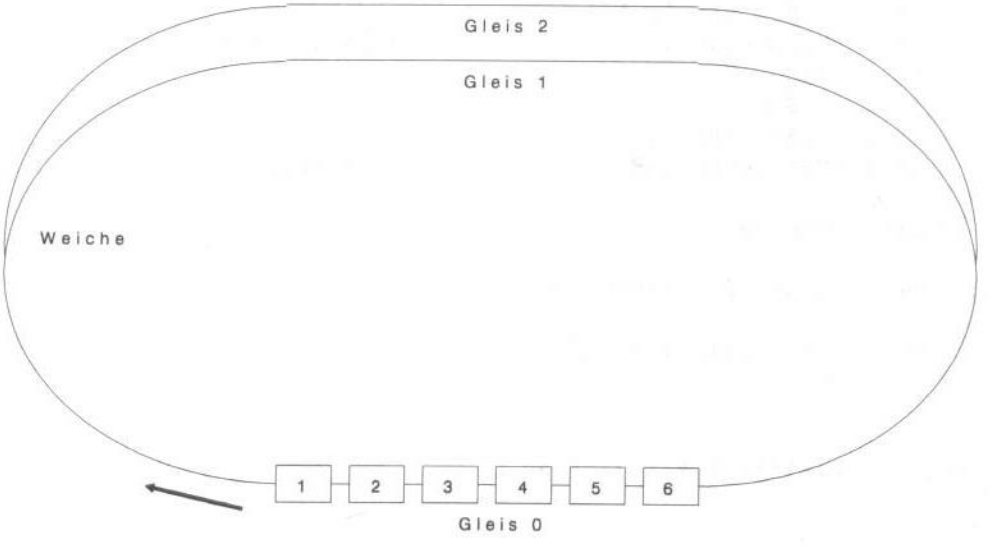
2.Phase
Die Loks fahren nun möglichst geordnet von den Rangiergleisen auf das Hauptgleis ein.



Wenn die Sortierung noch nicht vollständig ist, wird beide Phase noch einmal durchlaufen.



Es dürfte relativ schnell klar sein, daß mit diesem Algorithmus jede beliebige Anzahl von Loks geordnet werden kann. Die Grenzen bestehen natürlich in den verfügbaren Gleislängen und der faktischen Endlichkeit eines jeden Hobbybudgets.



Listing of **LOKSORT2.PAS**

```
PROGRAM LokSort;

  CONST SortGleisZahl = 2;  (* Anzahl der Gleise, über die sortiert wird *)
        MaxLokZahl = 5;

  TYPE Liste = ^ListenAnfang;
       ListenAnfang = RECORD
                        LokNr : 1..MaxLokZahl;
                        Rest : Liste
                      END;

       GleisNummer = 0..SortGleisZahl; (* Gleis 0 ist das Ausgangsgleis *)

  VAR   Gleis : ARRAY[0..SortGleisZahl] OF Liste;
        GleisKont : ARRAY[0..SortGleisZahl] OF
                      RECORD
                        Position : 0..MaxLokZahl;
                        Ende : ARRAY[1..MaxLokZahl] OF
                          RECORD Modul, Kontakt : BYTE END
                      END;
        LokZahl : 1..MaxLokZahl;
        Lok : ARRAY[1..MaxLokZahl] OF RECORD Nr,Prior,V : BYTE END;

  PROCEDURE SortLoks;
    (* Die Loks werden von Gleis 0 über die Sortiergleise geordnet *)
    VAR Schalter : 1..SortGleisZahl;
        LaufZahl : INTEGER;
        Letzte, Aktuelle : INTEGER;
        SortStatus : (nix,fast,fertig);

    PROCEDURE FahreBis(Nr,V : BYTE; Modul,Kontakt : BYTE);
      var RModule : ARRAY[1..4,1..16] of CHAR;

      procedure GetModule;
        const pause = 20;
        var c : char;
            b : byte absolute c;
         i,Nr : integer;
        begin
          FOR Nr:=1 TO 4 DO BEGIN
            delay(pause); write(aux,chr(Nr+192));
            read(aux,c);
            for i:=1 to 8 do
              if b and ($80 shr (i-1)) > 0
                then RModule[Nr,i]:='1' else RModule[Nr,i]:='0';
            read(aux,c);
            for i:=1 to 8 do
              if b and ($80 shr (i-1)) > 0
                then RModule[Nr,i+8]:='1' else RModule[Nr,i+8]:='0'
          END (* FOR *)
        end; (* GetModule *)

      procedure SendLok(Nr, Funktion : BYTE);
        begin
          write(aux,chr(Funktion),chr(Nr)); delay(100);
        end; (* SendLok *)

      BEGIN
        writeln; writeln(Nr:10,V:10); readln;
        SendLok(Nr,V);
        REPEAT
```

```
      GetModule
    UNTIL RModule[Modul,Kontakt]='1';
    SendLok(Nr,0)
  END; (* FahreBis *)

PROCEDURE SchalteWeicheAuf(Nr : GleisNummer);
  type Magn = record
                Nr : byte;
                Stellung : char;
              end;

  var M : Magn;

  procedure SendMagnet(M : Magn);
    begin
      with M do begin
        if (Nr>0) and (upcase(Stellung) in ['R','G']) then begin
          case upcase(Stellung) of
            'R' : write(aux,#34,chr(Nr));
            'G' : write(aux,#33,chr(Nr));
          end; (* case *)
          delay(500);
          write(aux,#32)
        end (* if *)
      end (* with *)
    end; (* SendMagnet *)

 BEGIN (* SchalteWeicheAuf *)
   M.Nr:=4;
   CASE Nr OF
     1 : M.Stellung:='R';
     2 : M.Stellung:='G';
   END;
   SendMagnet(M)
 END; (* SchalteWeicheAuf *)

PROCEDURE Fahren(von,nach : GleisNummer);
  (* Hängt das Element p an die Liste L (umständlich!) *)
  VAR p,q : Liste;
  BEGIN
    q:=Gleis[von]^.Rest;
    WriteLn('Lok ',Gleis[von]^.LokNr,' von Gleis ',von,
            ' fährt auf Gleis ',nach,'.');
    WITH GleisKont[nach] DO BEGIN
      Position:=Position+1;
      WITH Lok[Gleis[von]^.LokNr] DO WITH Ende[Position] DO
        FahreBis(Nr,V,Modul,Kontakt);
    END;
    IF Gleis[nach]=NIL
    THEN BEGIN Gleis[nach]:=Gleis[von]; p:=Gleis[nach] END
    ELSE BEGIN
           p:=Gleis[nach]; WHILE p^.Rest<>NIL DO p:=p^.Rest;
           p^.Rest:=Gleis[von]; p:=p^.Rest
         END;
    Gleis[von]:=q; p^.Rest:=NIL
  END; (* Fahren *)

PROCEDURE Zerlege;
  BEGIN
    FOR Schalter:=1 TO SortGleisZahl DO BEGIN
      Gleis[Schalter]:=NIL; GleisKont[Schalter].Position:=0;
    END;
    SchalteWeicheAuf(1);
```

```
      Schalter:=1; Letzte:=Gleis[0]^.LokNr; Fahren(0,1); LaufZahl:=1;
      WHILE Gleis[0]<>NIL DO BEGIN
        Aktuelle:=Gleis[0]^.LokNr;
        IF Lok[Aktuelle].Prior<Lok[Letzte].Prior THEN BEGIN
          LaufZahl:=succ(LaufZahl);
          IF Schalter=SortGleisZahl
            THEN Schalter:=1 ELSE Schalter:=succ(Schalter);
          SchalteWeicheAuf(Schalter);
        END; (* IF *)
        Fahren(0,Schalter); Letzte:=Aktuelle
      END; (* WHILE *)
      IF LaufZahl=1 THEN SortStatus:=fertig ELSE
      IF LaufZahl<=SortGleisZahl THEN SortStatus:=fast
      ELSE SortStatus:=nix
    END; (* Zerlege *)

  PROCEDURE Mische;
    VAR Schalter : GleisNummer;
        Letzte : 1..MaxLokZahl;

    FUNCTION Erste : GleisNummer;
      VAR i,t : GleisNummer;
      BEGIN
        t:=0; FOR i:=1 TO SortGleisZahl DO IF Gleis[i]<>NIL THEN t:=i;
        IF t<>0 THEN
          FOR i:=1 TO SortGleisZahl DO IF Gleis[i]<>NIL THEN
            IF Lok[Gleis[i]^.LokNr].Prior<Lok[Gleis[t]^.LokNr].Prior
              THEN t:=i;
        Erste:=t
      END; (* Erste *)

    FUNCTION Naechste : GleisNummer;
      VAR i,t : GleisNummer;
      BEGIN
        t:=0;
        FOR i:=1 TO SortGleisZahl DO IF Gleis[i]<>NIL THEN
          IF Lok[Gleis[i]^.LokNr].Prior>=Lok[Letzte].Prior THEN
            IF t=0 THEN t:=i ELSE
            IF Lok[Gleis[i]^.LokNr].Prior<Lok[Gleis[t]^.LokNr].Prior
            THEN t:=i;
        IF t=0 THEN t:=Erste;
        Naechste:=t
      END; (* Naechste *)

  BEGIN (* Mische *)
    Schalter:=Erste; GleisKont[0].Position:=0;
    WHILE Schalter<>0 DO BEGIN
      Letzte:=Gleis[Schalter]^.LokNr;
      Fahren(Schalter,0); Schalter:=Naechste
    END; (* WHILE *)
    IF SortStatus=fast THEN SortStatus:=fertig
  END; (* Mische *)

BEGIN
  REPEAT
    Zerlege; WriteLn;
    Mische
  UNTIL SortStatus=fertig
END;

PROCEDURE InitLoks;
  VAR i : INTEGER;
      p : Liste;
```

```
    BEGIN
      ClrScr;
      Write('Anzahl der Loks: '); ReadLn(LokZahl);
      Write('Nummern der Loks in derz. Reihenfolge: ');
      FOR i:=1 TO LokZahl DO BEGIN
        readln(Lok[i].Nr,Lok[i].V);
      END;
      writeln; WriteLn('Vielen Dank!'); WriteLn;
      Gleis[0]:=NIL;
      FOR i:=LokZahl DOWNTO 1 DO BEGIN
        NEW(p); p^.LokNr:=i; p^.Rest:=Gleis[0]; Gleis[0]:=p
      END (* FOR *)
    END; (* InitLoks *)

  PROCEDURE LiesReihenFolge;
    VAR i,t : INTEGER;
        Loks : SET OF 1..MaxLokZahl;
    BEGIN
      Loks:=[]; FOR i:=1 TO LokZahl DO Loks:=Loks+[i];
      Write('Zu sortierende Reihenfolge: ');
      REPEAT
        ReadLn(t);
        IF t IN Loks THEN BEGIN
          Lok[t].Prior:=i; Loks:=Loks-[t]
        END
      UNTIL Loks=[];
      WriteLn
    END; (* LiesReihenFolge *)

  PROCEDURE SchreibAusgang;
    VAR p : Liste;
    BEGIN
      ClrScr;
      write('Jetzt stehen die Loks in der Folge: ');
      p:=Gleis[0];
      WHILE p<>NIL DO BEGIN write(p^.LokNr:3); p:=p^.Rest END;
      WriteLn
    END; (* SchreibAusgang *)

  PROCEDURE InitGleise;
    BEGIN
      GleisKont[0].Ende[1].Modul:=1; GleisKont[0].Ende[1].Kontakt:=6;
      GleisKont[0].Ende[2].Modul:=2; GleisKont[0].Ende[2].Kontakt:=2;
      GleisKont[0].Ende[3].Modul:=2; GleisKont[0].Ende[3].Kontakt:=7;
      GleisKont[0].Ende[4].Modul:=3; GleisKont[0].Ende[4].Kontakt:=2;
      GleisKont[0].Ende[5].Modul:=0; GleisKont[0].Ende[5].Kontakt:=0;
      GleisKont[1].Ende[1].Modul:=4; GleisKont[1].Ende[1].Kontakt:=13;
      GleisKont[1].Ende[2].Modul:=4; GleisKont[1].Ende[2].Kontakt:=7;
      GleisKont[1].Ende[3].Modul:=4; GleisKont[1].Ende[3].Kontakt:=14;
      GleisKont[1].Ende[4].Modul:=0; GleisKont[1].Ende[4].Kontakt:=0;
      GleisKont[1].Ende[5].Modul:=0; GleisKont[1].Ende[5].Kontakt:=0;
      GleisKont[2].Ende[1].Modul:=3; GleisKont[2].Ende[1].Kontakt:=16;
      GleisKont[2].Ende[2].Modul:=4; GleisKont[2].Ende[2].Kontakt:=11;
      GleisKont[2].Ende[3].Modul:=4; GleisKont[2].Ende[3].Kontakt:=1;
      GleisKont[2].Ende[4].Modul:=0; GleisKont[2].Ende[4].Kontakt:=0;
      GleisKont[2].Ende[5].Modul:=0; GleisKont[2].Ende[5].Kontakt:=0;
    END; (* InitGleise *)

procedure InitRS232;
var Regs:record ax,bx,cx,dx,bp,si,di,ds,es,flags : integer end;
  begin
    Regs.ax:=$00a7; Regs.dx:=0; (* com1 *)
    intr($14,Regs);
```

```
  (* mode com1:2400,n,8,2 zum Einstellen der Schnittstellenparameter aufrufen *)
end;

BEGIN (* LokSort *)
  InitRS232;
  InitLoks;
  REPEAT
    SchreibAusgang;
    LiesReihenFolge;
    SortLoks;
    ReadLn(TRM);
  UNTIL FALSE
END.  (* LokSort *)
```

Vervollständigen Sie noch heute Ihre Software-Bibliothek!

Bestellen Sie die Ihnen noch fehlenden CHIP-SPECIALS. Mit den nebenstehenden Karten geht's ganz leicht. Einfach ausfüllen und ab die Post!

1 Bestellkarte für weitere CHIP-SPECIALS

☐ **Ja,** senden Sie mir bitte die angekreuzten SPECIALS zu den genannten Preisen zuzüglich Versandkostenanteil DM 3,50 im Inland. (Versandkostenanteil für das Ausland DM 6,–).

Ich bezahle erst, wenn ich Ihre Rechnung erhalten habe.

Datum

Unterschrift

Bitte genaue Anschrift auf der Rückseite angeben.

Anzahl	Titel	Best.-Nr.	DM/Stck.
	Amiga, 1. Ausgabe	0290	28,—
	Amiga, 2. Ausgabe	0410	28,—
	Amiga, 3. Ausgabe	0550	28,—
	Amiga, 4. Ausgabe	0640	28,—
	Atari ST 1	0230	28,—
	Atari ST 2	0390	28,—
	Atari ST 3	0470	28,—
	Atari ST 4	0490	28,—
	C 64/128 „Sportspiele"	0380	24,—
	MS-DOS, Ausgabe 1	0949	49,—
	MS-DOS, Ausgabe 2	0951	49,—
	Prozessor 68000	0590	28,—
	Computer steuert Märklin-Modellbahn	0953	19,—
	C-Einstieg	0440	28,—

1475

2 Wünsche an die Redaktion

Am vorliegenden ERSTEN MÄRKLIN-SPECIAL gefällt mir:

gut: _____

weniger: _____

nicht: _____

Folgendes Thema sollten Sie unbedingt im nächsten Heft abhandeln:

_____

CHIP SOFT

Bestellen Sie noch heute die Programme für Ihren Rechner

3 Bestellkarte für CHIP-SOFT Märklin, Ausgabe 1

Ja, senden Sie mir:

☐ 1 IBM-PC-Diskette mit Beispielprogrammen aus diesem Heft DM 29,—

Inlandspreise inkl. MwSt. + 3,50 DM Versandkostenanteil, 3,20 DM Nachnahmegebühr; Ausland: 6,– DM Versandkostenanteil plus Nachnahme.

Datum

Unterschrift

Bitte genaue Anschrift auf der Rückseite angeben.

1476

Name, Vorname

Straße/Postfach

PLZ/Ort

Bitte mit 60 Pfennig freimachen

Antwort

Vogel-Verlag
CHIP-SPECIAL
Leser-Service 735
Postfach 6740

D-8700 Würzburg 1

CHIP SPECIAL

Name, Vorname

Straße, Postfach

PLZ/Ort

Bitte mit 60 Pfennig freimachen

Antwort

Vogel-Verlag
Redaktion CHIP-SPECIAL
Schillerstr. 23 a

D-8000 München 2

Name, Vorname

Straße, Postfach

PLZ/Ort

Bitte mit 60 Pfennig freimachen

Antwort

CHIP-SHOP
Leser-Service 735
Vogel-Verlag
Postfach 6740

D-8700 Würzburg 1

CHIP SPECIAL

1. Auflage 1987

Best.-Nr. 0953

Redaktionsdirektor: Richard Kerler

Chefredakteur: Armin Schwarz
(verantwortlich für den Inhalt)

Stellvertretender Chefredakteur: Ulrich Kern

Redaktionsassistenz, Leserservice: Petra Eibicht

Layout: Karlheinz Dereser

Autoren dieser Ausgabe: Gerhard Bader, Frank Frauenhoffer, Ulrich Kern, Wolfgang Otternberg, Ullrich Thiemann

Gestaltung: Hans Kuh

Titelfoto: Studio Eins

Redaktion: Vogel-Verlag KG Würzburg, Redaktion CHIP-Special, Schillerstr. 23a, D-8000 München 2, Telefon (089) 514930, Telekopierer 535000, Telex 5216449

Verlag: Vogel-Verlag KG, Postfach 6740, D-8700 Würzburg 1, Tel. (0931) 4102-1, Telex 68883, Telefax (0931) 4102-529. Telegramme: CHIP-Würzburg

Verlagsdirektor: Dr. Andreas Kaiser

Anzeigenleiter: Friedrich Mangold, Würzburg (verantwortlich für Anzeigen)

Anzeigenservice: CHIP, Postfach 6740, 8700 Würzburg 1, Tel. (0931) 4102-1, Telex 68883, Michael Belgrad, Durchwahl 4102-433.

Vertriebsleitung: Axel Herbschleb, Würzburg

Vertrieb Handelsauflage: Vereinigte Motor-Verlage GmbH & Co. KG, Leuschnerstr. 1, D-7000 Stuttgart 1, Tel. (0711) 2043-1

Bezugsmöglichkeiten: Bestellungen nehmen der Verlag und alle Buchhandlungen im In- und Ausland entgegen. Sollte die Zeitschrift aus Gründen, die nicht vom Verlag zu vertreten sind, nicht geliefert werden können, besteht kein Anspruch auf Nachlieferung oder Erstattung vorausbezahlter Bezugsgelder.

Bankverbindungen Vogel-Verlag:
Dresdner Bank AG, Würzburg
(BLZ 79080052) 3148890000,
Bay. Vereinsbank AG, Würzburg
(BLZ 79020076) 2506173,
Kreissparkasse Würzburg
(BLZ 79050130) 17400,
Postscheckkonto Nürnberg
(BLZ 76010085) 9991-853
Ausland: Postscheckkonto Zürich
8047064,
Niederlande 2662395
Banque Veuve Morin-Pons, Paris
155410314

Gesamtherstellung und Versand: VOGEL-DRUCK WÜRZBURG, Max-Planck-Str. 7/9, D-8700 Würzburg

CIP-Kurztitelaufnahme der Deutschen Bibliothek

Bader, Gerhard
Computer steuert Märklin-Modellbahn: Programme für Atari ST, Amiga, Commodore 64/128, IBM PC u. Kompatible u. alle Computer mit serieller Schnittstelle/ [Autor dieser Ausg.: Gerhard Bader]. - 1. Aufl. - Würzburg: Vogel, 1987
(Chip special)
ISBN 3-8023-0953-7
NE: HST

In dieser Ausgabe

Autor: Gerhard Bader

Grundwissen	**Start**
Digital-Eisenbahn	**Märklin Digital HO**
Digital-Eisenbahn	**Die Komponenten des Märklin-Digital-HO-System**
Praxis	**Anwendungsspiele**
Speicherprogrammierbare Steuerungen	**SPS**
Möglichkeiten	**Die Vorteile des Digital-Betriebs**
Schnittstelle	**Das Märklin-Interface 6050**
Steuern und Regel	**Grundlegende Steuerungen**
Ordnen	**Sortieren in Sekundenschnelle**
Künstliche Intelligenz	**Turbo-Prolog stellt die Weichen**
Fahrplan	**Fahrwegoptimierung**
Steuerungs-Programm	**ESPV3 steuert die Modellbahn**
train-ing	**Spielend lernen**
Gleisbild-Editor	**Ein interaktives Universalprogramm**
Sortieren	**Loksort**

4 003634 009537